Information Security

Business Continuity & Desaster Recovery - das Planspiel in Containern

Diplomarbeit

zur Erlangung des akademischen Grades

Diplom-Ingenieur/in

eingereicht von

Stefan Machherndl, BSc

is191812

im Rahmen des

Studienganges Information Security an der Fachhochschule St. Pölten

Betreuung

Betreuer/in: Dipl.-Ing. Dipl.-Ing. Christoph Lang-Muhr, BSc

Mitwirkung: -

St. Pölten, 5. Oktober 2021

(Unterschrift Verfasser/in) (Unterschrift Betreuer/in)

Ehrenwörtliche Erklärung

Ich versichere, dass

- ich diese Diplomarbeit selbständig verfasst, andere als die angegebenen Quellen und Hilfsmittel nicht benutzt und mich sonst keiner unerlaubten Hilfe bedient habe.
- ich dieses Diplomarbeitsthema bisher weder im Inland noch im Ausland einem Begutachter/einer Begutachterin zur Beurteilung oder in irgendeiner Form als Prüfungsarbeit vorgelegt habe.
- diese Arbeit mit der vom Begutachter/von der Begutachterin beurteilten Arbeit übereinstimmt.

Der Studierende/Absolvent räumt der FH St. Pölten das Recht ein, die Diplomarbeit für Lehre- und Forschungstätigkeiten zu verwenden und damit zu werben (z.B. bei der Projektevernissage, in Publikationen, auf der Homepage), wobei der Absolvent als Urheber zu nennen ist. Jegliche kommerzielle Verwertung/Nutzung bedarf einer weiteren Vereinbarung zwischen dem Studierenden/Absolventen und der FH St. Pölten.

Ort, Datum *Unterschrift*

Kurzfassung

Diese Arbeit befasst sich mit der Umsetzung einer automatisierten Infrastruktur für ein Planspiel. Im Zuge dieses Planspiels wird der Neuaufbau einer digitalen Kommunikationsinfrastruktur simuliert, diese Infrastruktur soll zeitgemäße und sichere Technologien verwenden und im technischen Sinne möglichst realistisch sein.

Die Arbeit beschreibt die Entwicklung und Umsetzung eines Prototyps der den in der Arbeit gestellten Anforderungen entspricht. Im Zuge dessen, werden verschiedene Technologien in Betracht gezogen und die dafür geeignetsten ausgewählt. Der entwickelte Prototyp wurde bereits in zwei Durchläufen des Planspiels verwendet und durch diese auch weiter verbessert.

Abstract

This paper deals with the implementation of an automated infrastructure for a simulation game. In the course of this simulation game, the rebuilding of a digital communication infrastructure is simulated. This built infrastructure should use up-to-date and secure technologies and should be as realistic as possible in a technical sense.

The paper describes the development and implementation of a prototype that meets the requirements described in the paper. Different technologies are considered and the most suitable ones are selected. The developed prototype has already been used in two runs of the simulation game and has been further improved by them.

Inhaltsverzeichnis

1. Einleitung

Im Zuge des Bachelorstudiums IT-Security an der Fachhochschule St.Pölten wird ein Planspiel zur Festigung der gelernten Kompetenzen durchgeführt. Das Planspiel trägt den Namen Business Continuity and Desaster Recovery (BCDR) . Dabei werden die Studierenden vor die Aufgabe gestellt den Wiederaufbau der weltweiten digitalen Kommunikationsinfrastruktur (des Internets), die in diesem Szenario vollständig zerstört wurde, durchzuführen. Dazu sollen zeitgemäße und sichere Technologien verwendet werden. [1]

1.1. Beschreibung des Planspiels

Zu Beginn des Planspiels werden die Studierenden darüber informiert, dass die gesamte Kommunikationsinfrastruktur durch einen Sonnensturm zerstört wurde. Dieses Szenario soll den Studierenden die Risiken einer vollständig vernetzten Welt, die durch natürliche Phänomene zum Stopp gezwungen wird, vor Augen halten. [1]

Das Ziel des Planspiels ist, dass die Teilnehmer/innen gemeinsam die zerstörte Kommunikationsinfrastruktur wiederaufbauen. Dazu belegen die Studierenden vordefinierte Unternehmen, welche mit der Umsetzung von Teilgebieten des Wiederaufbaus der digitalen Kommunikationsinfrastruktur beauftragt werden. Folgende Funktionalitäten müssen die Gruppen im Zuge des Wiederaufbaus bereitstellen:

- Planung einer neuen Netzwerktopologie und entsprechendes Routing
- Implementation von Webserver die im neu implementierten Netzwerk erreichbar sind
- Implementation einer Mail-Infrastruktur
- Implementation eines eigenständigen Domain Name Systems (DNS)
- Implementation einer Public-Key-Infrastruktur (PKI)

Diese genannten Anforderungen werden automatisch von einem Monitoringsystem auf richtige Funktionalität überprüft. Während des Neuaufbaus sollen nur Technologien verwendet werden die aus aktueller Sicht sicher sind und auch in Zukunft verwendet werden (z.B. keine Verwendung von Internet Protocol Version 4 (IPv4), nur Internet Protocol Version 6 (IPv6) darf verwendet werden). Um den Studierenden Fortschritt im

Spielgeschehen zu vermitteln, ist das Planspiel in Ticks und Phasen eingeteilt. Ticks beschreiben einen gewissen Zeitabschnitt und Phasen beschreiben den Fortschritt der einzelnen Gruppen. Zusätzlich wird jeden Tick eine Zahl an weltweiten Opfern berechnet, im Szenario des Planspiels ist das Bedürfnis an verfügbarer Kommunikationsinfrastruktur so hoch, dass bei nicht erfüllen dieses Bedürfnisses Menschen ihr Leben verlieren. Die berechnete Zahl an Opfern ist vom Fortschritt der Gruppen abhängig, erfüllen alle Gruppen alle gestellten Anforderungen ist die Zahl der Opfer 0. Je weniger Anforderungen erfüllt sind, desto höher ist die Zahl der Opfer. Das Ziel der Studierenden ist es die Kommunikationsinfrastruktur wieder aufzubauen und dabei die Opferzahl möglichst niedrig zu halten. Die Opferzahl am Ende des Spiels wird als Vergleichswert der verschiedenen Jahrgänge verwendet. [1]

Die Definition von Ticks und Phasen im Planspiel hat sich durch Weiterentwicklungen zu der Definition im Originaldokument [1] verändert. Aktuell werden Ticks zur Abrechnung, dem Eintreten von Ausfällen und zur Erhebung des Gesamtzwischenstandes (Opferzahl) verwendet. Ticks werden regelmäßig und nicht abhängig von Phasen durchgeführt. Phasen dokumentieren den Fortschritt einer Gruppe. Je höher die Phase einer Gruppe, desto mehr Funktionalitäten stellt die Gruppe zur Verfügung, auch wird dadurch der Gewinn, der während der Abrechnung am Ende jedes Ticks berechnet wird, erhöht. Befinden sich alle Gruppen in der höchsten Phase, so steigt die Opferzahl nicht an, ansonsten wird die Opferzahl anhand der erreichten Phasen aller Gruppen berechnet. [1]

Die technische Umsetzung der geforderten Funktionalitäten wird durch das automatisierte Monitoring der Router und Hosts im Netzwerk überprüft, dieses Monitoringsystem soll auch weiter verwendet werden. Die folgenden technischen Komponenten sind im Planspiel enthalten:

- **Desktop PC:** Dient als Konfigurationsterminal der Router, ohne ihn können die Router nicht konfiguriert werden.
- **Small-, Medium-, Large Server, Hypervisor:** Auf diesen Geräten werden die Dienste wie Mail, DNS, Web Server betrieben die jedes Unternehmen braucht.
- **National Line, International Line, Intercontinental Line:** Netzwerk-Verbindungen zwischen den Spielern und deren Standorten.
- **Router, Enterprise:** Router um die unterschiedlichen Netzwerke zu verknüpfen.
- **SLA Silver, SLA Gold:** Service Level Agreements um die Ausfallsicherheit der Komponenten zu erhöhen. [1]

Um die Komponenten zu virtualisieren wurde für Server die Software VMware Workstation [2] und für

Router GNS3 [3] verwendet. [1]

Die im original Dokument [1] erwähnte Umsetzung des Modelles durch ein Microsoft Excel Sheet ist nicht mehr aktuell. Darin wurde eine Liste der verfügbaren Hardware, die ausgefallene Hardware, der Kontostand und der Fortschritt jeder Gruppe dokumentiert, jeden Tick wurden darin die neuen Kontostände berechnet und die neu ausgefallene Hardware bestimmt. Diese im Microsoft Excel Sheet erledigten Funktionen wurden durch Studierendenprojekte in einer Webapplikation realisiert, auf welches die jeweiligen Gruppen als auch die Spielleitung Zugriff hat. Durch die Implementierung in einer Webapplikation wird das Sheet nicht mehr verwendet. Die Gruppen können über die Spieloberfläche oder Webapplikation neue Hardware „kaufen“ und vorhandene verwalten (bereitgestellte Funktionalitäten eintragen, verkaufen). Es ist auch eine Schnittstelle zwischen dieser Spieloberfläche und dem Monitoringsystem vorhanden, dadurch kann das Monitoringsystem neue bereitgestellte Funktionalitäten übernehmen und ebenfalls überprüfen.

Die Implementation der Spieloberfläche und der Schnittstelle zum Monitoringsystem wurde nicht im Rahmen diese Arbeit durchgeführt. Diese beiden Komponenten werden im Rahmen der Arbeit aktualisiert, um die geforderten neuen Funktionalitäten bereitzustellen.
Genauere Informationen zum Spielablauf und zur geforderten Funktionaliät, die die Gruppen im Zuge des Planspiels implementieren sollen, werden in internen Dokumenten bereitgestellt. Das grundlegende Dokument zur Erklärung des Planspiels ist mit Anhang A dieser Arbeit beigefügt.

1.2. Forschungsfrage

Dieses Planspiel wurde bisher mit der in den Netzwerklaboren der Fachhochschule St.Pölten zur Verfügung stehenden Infrastruktur durchgeführt, um eine möglichst realitätsnahe Umgebung zu schaffen. Um diese Infrastruktur zu nutzen mussten die Studierenden vor Ort sein. Die Durchführung des Planspiels mit Studierenden welchen es nicht möglich war vor Ort zu sein, konnte nur mit großen Aufwand umgesetzt werden. Auf die verwendeten Virtuellen Maschinen (VM) konnte jeweils ein/e Student/in zugreifen, eine Konfiguration durch andere Studierende war nicht möglich. Die Herstellung der Netzwerkverbindung zwischen den verwendeten Komponenten war nur vor Ort möglich, da diese das Umstecken von Netzwerkkabeln beinhaltet. Somit waren Gruppen, in welchen Studierende nicht vor Ort waren, benachteiligt.

Eine Idee zur Weiterentwicklung des Planspiels beinhaltet, dass nicht nur Studierende der Fachhochschule

St.Pölten in das Planspiel involviert werden. Es sollen auch Studierenden anderer Universitäten beteiligt werden, um das Planspiel realistischer zu gestalten und internationale Zusammenarbeit zu fördern. Der Gedankengang dazu ist, dass bei einem Wiederaufbau der weltweiten digitalen Kommunikationsinfrastruktur nicht nur technische Probleme auftreten, sondern auch geographische Distanz (Zeitverschiebungen) und sprachliche Barrieren eine Rolle spielen.
Um diese Idee umzusetzen, muss die dafür notwendige Infrastruktur auch aus der Ferne einfach zugänglich sein, um gleichmäßige Verhältnisse zwischen Studierenden vor Ort und Studierenden in der Ferne zu schaffen. Es soll keine Vor- oder Nachteile für Studierende an verschiedenen geographischen Punkten geben.

Ein weiteres Ziel zur Weiterentwicklung des Planspiels ist, die Infrastruktur zu automatisieren. Das wird vor allem die Spielgeschwindigkeit in den ersten Stunden erhöhen, da die Studierenden hier die meiste Zeit mit dem vorbereiten von Virtuelle Maschine (VM)s und dem Kennenlernen der verwendeten Software zur Emulation von Netzwerken (GNS3) [3] beschäftigt waren. Außerdem wird die automatisierte Infrastruktur das Schummeln der Studierenden besser unterbinden, da zuvor auf das einhalten der Regeln bei Ausfall einer Komponente vertraut werden musste. Mit automatisierter Infrastruktur können die Komponenten zielgerichtet, um einen Ausfall zu simulieren, abgeschaltet werden.

Aus diesen Punkten haben sich folgende Ziele entwickelt:
Die Infrastruktur soll vollständig virtualisiert werden. Die für das Spiel notwendigen Komponenten sollen auf der bereits vorhandenen Spieloberfläche „gekauft“ und verwaltet werden können. Zugriff auf diese virtualisierte Hardware soll auch über die Spieloberfläche erfolgen. Damit ist sichergestellt, dass sowohl Studenten die vor Ort sind, als auch Studenten die sich aus der Ferne am Planspiel beteiligen mit der selben Umgebung arbeiten können. Da nach erfolgreicher Umsetzung keine Infrastruktur aus den Netzwerklaboren verwendet wird, und alle Komponenten über die Spieloberfläche konfiguriert und verwendet werden können.

Aus diesen Zielen sind folgende Forschungsfragen entstanden:

„Welche Technologien können zur Automatisierung und Virtualisierung einer Netzwerkinfrastruktur verwendet werden?“

„Welche Technologien sind nötig um die benötigte Infrastruktur des Planspiels „ Business Continuity & Desaster Recovery“ vollständig zu automatisieren und zu virtualisieren und sinnvoll in das Planspiel einzubinden?“

2. Anforderungen

Grundsätzliche Anforderung an die virtualisierte Infrastruktur ist, dass diese sehr ressourcenschonend betrieben werden kann. Da im Zuge des Planspiels erfahrungsgemäß eine große Anzahl an Server und Router benötigt werden, diese jedoch keine leistungsintensiven Aufgaben durchführen müssen, da nur die Studierenden das neu aufgebaute Netzwerk benutzen. Aus diesem Grund soll es möglich sein die Server und Router mit möglichst geringem Ressourcenverbrauch (CPU-Last und Arbeitsspeicherverbrauch) zu betreiben. Aus Erfahrungswerten werden ca. 20 Server und 20 Router im Laufe des Planspiels verwendet. Um das Spiel auch mit größeren Gruppen durchführen zu können, sollen mindestens 40 Server und 40 Router zur Verfügung gestellt werden können. Zur Umsetzung steht ein Server mit 48 CPU Kernen (auf 2 Sockets) mit jeweils 2,4 GHz Taktgeschwindigkeit und 160 GB Arbeitsspeicher zur Verfügung.

Ein weiterer wichtiger Punkt ist es, die bereits vorhandene Spieloberfläche und das Monitoringsystem in die neue Infrastruktur einzubinden. Falls eine Neuimplementation der Spieloberfläche und des Monitoringsystems nötig wäre, um diese in die neue Infrastruktur einzubinden, kann diese nicht im Zuge dieser Arbeit durchgeführt werden. Da eine Neuimplementation dieser Komponenten den Rahmen dieser Arbeit übersteigen würde. Ein angedachter Aufbau der Komponenten wird in Abbildung 2.1 gezeigt. Das Monitoringsystem soll sich im aufgebauten Netzwerk des Planspiels befinden. Spieloberfläche und die Plattform zur Steuerung der virtuellen Infrastruktur (Virtualisierungsplattform) sollen aus dem Labornetzwerk erreichbar sein. Zwischen Spieloberfläche und Monitoringsystem soll ein Kommunikationskanal bestehen, um Informationen zum aktuellen Spielgeschehen auszutauschen.

Die Spieloberfläche muss mithilfe einer Schnittstelle die virtualisierte Infrastruktur steuern bzw. verändern können. Somit soll eine vollständige Automatisierung der Infrastruktur erreicht werden. Folgende Beispiele sollen das verdeutlichen: Eine Gruppe kauft in der Spieloberfläche einen neuen Server, dieser wird nach erfolgreichem Kauf automatisch provisioniert. Nachdem der Server erfolgreich erstellt wurde, erhält die Gruppe automatisch Zugriff auf diesen Server, um diesen entsprechend zu konfigurieren. Wenn diese Gruppe den

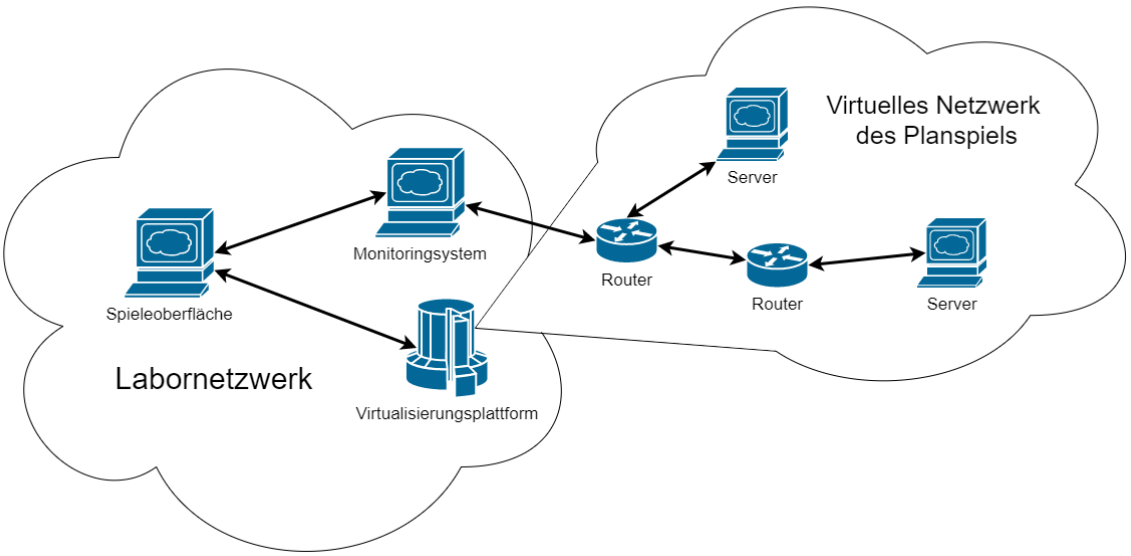


Abbildung 2.1.: Angedachter Aufbau von Spieloberfläche, Virtualisierungsplattform, Monitoringsystem und virtuellem Spielnetzwerk

Server an das im Planspiel entstehende Netzwerk anbinden will, geschieht das über die Spieloberfläche. Die Gruppe kann hier einstellen mit welcher Line (Netzwerkverbindung zwischen mehreren Server/Routern) der Server verbunden werden soll, diese Einstellung wird voll automatisch von der virtualisierten Infrastrukur angewendet.
Ein weiteres Szenario ist der Ausfall einer Komponente, Ausfälle werden durch die Spieloberfläche bei jedem Tick zufällig ausgewählt. Sobald ein solcher Ausfall bestimmt wurde, wird die entsprechende Komponente in der virtuellen Infrastruktur automatisch ausgeschaltet bzw. betriebsunfähig geschaltet. Die betroffene Gruppe wird über diesen Ausfall durch die Spieloberfläche informiert.

Um die Entwicklung der virtuellen Infrastruktur und die Einbindung der bereits vorhandenen Spielelemente zu erleichtern, ist eine bereits vorhandene Plattform (Virtualisierungsplattform) von Vorteil, dadurch können zumindest Teile der bereits angesprochenen Schnittstelle zur Spieloberfläche realisiert werden. Eine eigenständige Entwicklung einer vollständigen Schnittstelle zwischen Infrastruktur und Spieloberfläche wird die Qualität dieser verringern und die Gesamtentwicklungszeit deutlich erhöhen. Auch kann die Verwendung einer Virtualisierungsplattform den Vorteil bringen, zusätzliche Features dieser Plattform zu nutzen und möglicherweise neue Spielmechaniken zu etablieren.

Die Verwendung von Open Source Software [4] wird angestrebt, um zu zeigen, dass Projekte dieser Art

auch mit Open Source Software durchgeführt werden können, und nicht nur proprietäre Software verwendet werden muss. Auch wird dadurch erwartet, mehrere offene Schnittstellen zu erhalten und so die Komponenten der Infrastruktur einfacher miteinander zu verbinden. Ein weiterer Vorteil ist die Möglichkeit, die verwendete Software anzupassen und so Funktionalitäten zu verändern um eigene Anforderungen zu erfüllen. Zusätzlich können durch Verwendung von Open Source Software Lizenzkosten gespart werden.

2.1. Netzwerktechnische Anforderungen

Wie bereits erläutert, soll die Netzwerkebene der Infrastruktur automatisch generiert und verändert werden.

Dafür müssen Netzwerkverbindungen automatisch erstellt werden können, diese Verbindungen müssen von bereits vorhandene Netzwerkinfrastruktur getrennt werden. Ein „Ausbruch“ aus dem Netzwerk des Planspiels soll nicht, oder nur kontrolliert, möglich sein.

Die Netzwerkverbindungen zwischen Komponenten des Planspiels müssen dynamisch hergestellt und auch wieder getrennt werden können. Diese Änderungen müssen möglichst schnell durchgeführt werden. Ziel ist es jede Änderung innerhalb von fünf Minuten automatisch umzusetzen. Diese Zeitspanne wurde gewählt, um eine Störung des Spielverlaufs so klein wie möglich zu halten. Jede Komponente muss mit jeder anderen Komponente auf Netzwerkebene verbunden werden können, um eine uneingeschränkte Entwicklung des Planspiels und den Gruppen freie Wahl in der Planung ihrer Netzwerktopologie zu ermöglichen.

Der Ausfall einer Netzwerkverbindung muss simuliert werden können, um essentielle Aspekte des Planspiels umzusetzen. Um einen solchen Ausfall möglichst realistisch zu gestalten, sollten die Netzwerkverbindungen von den Komponenten trotzdem erkannt werden, jedoch muss die Datenübertragung unterbunden werden. Als Beispiel kann hier ein bei Bauarbeiten beschädigtes oder durchgeschnittenes Kabel dienen.

2.2. Anforderungen an virtualisierte Server/Router

Da im Zuge des Planspiels eine hohe Anzahl an Server und Router virtualisiert werden, wird besonders der Ressourcenverbrauch betrachtet, die Server und Router sollen mit möglichst wenigen Ressourcen betrieben werden. Dabei ist zu beachten, dass keine großen Nutzerzahlen in dem von den Gruppen entworfenen Netzwerk zu erwarten sind, da nur die Teilnehmer/innen des Planspiels und das verwendete Monitoringsystem

auf dieses Netzwerk Zugriff haben.

Im Verlauf des Planspiels müssen die Gruppen verschiedene Funktionalitäten realisieren, um das zu ermöglichen soll die dazu benötigte Software auf den Servern und Routern ohne Probleme ausführbar und ohne großer Einschränkungen konfigurierbar sein. Um die Konfiguration möglichst realistisch zu gestalten soll Software verwendet werden, die auch in realen Kommunikationsnetzwerken verwendet wird.

Um die Server und Router zu konfigurieren, muss den Gruppen Zugriff auf diese bereitgestellt werden. Eine Gruppe darf dabei nur Zugriff auf ihre eigenen Komponenten erhalten. Der Zugriff auf die Komponenten soll, wenn möglich, über die bestehende Spieleoberfläche erfolgen. Die Verwendung von zusätzlicher Software, um einen Zugriff auf die Komponenten zu ermöglichen, wird vermieden.

Damit der Ausfall eines Servers oder eines Routers simuliert werden kann, müssen diese automatisch gestartet und gestoppt werden können. Das Starten und Stoppen der Server und Router soll möglichst wenig Zeit beanspruchen, um den Spielverlauf nicht zu stören. Auch hier wird ein Intervall von unter fünf Minuten angestrebt, in welchem die Änderungen umgesetzt werden sollen.

2.2.1. Anforderungen an Routing Software

Im Bachelorstudiengang IT-Security werden bereits einige Übungen mit dem virtuellen Router Cisco CSR 1000V [5] durchgeführt. Da dieser mindestens 4GiB Arbeitsspeicher benötigt, kann er im Planspiel nicht eingesetzt werden. Aus Erfahrungswerten werden ca. 20 Server und Router benötigt, dadurch liegt der benötigte Arbeitsspeicher nur durch Router bei ca. 80GiB. Ein weiteres Skalieren des Planspiels ist dadurch nur sehr eingeschränkt und mit hohen Hardwarekosten möglich.
Auf Grund dessen soll eine Routing Software eingesetzt werden, die auch mit wesentlich geringerem Ressourcenverbrauch stabil betrieben werden kann. Die verwendete Routing Software sollte jedoch Ähnlichkeiten zum bereits bekanntem Cisco IOS XE [6] aufweisen um den Studierenden die Anwendung zu erleichtern.

3. Stand der Forschung

Der Einsatz von virtuellen Netzwerk- und Serverinfrastrukturen in der Lehre wird bereits von einigen Universitäten erforscht. Netzwerkinfrastrukturen können durch Simulation oder Emulation realisiert werden [7]. Da sich Simulationen wie z.B. Cisco Packet Tracer [8] oft unterschiedlich zu realen Systemen verhalten, wie in [9] gezeigt. Soll nur Software, welche Emulation ermöglicht, für diese Arbeit verwendet werden. Der Vorteil der besseren Skalierbarkeit von Container basierter Emulation wird in [9] beschrieben, der Fokus dieser Arbeit wurde jedoch auf die Reproduzierbarkeit von Testszenarien in Forschungsarbeiten gelegt. In [7] werden die Netzwerk-Emulation-Tools Cisco Virtual Internet Routing Lab (VIRL), GNS3 [3] und Mininet [10] für den Einsatz in der Lehre evaluiert. Ein Vergleich dieser, und dem Nachfolger von VIRL - Cisco Modeling Labs (CML) [11], ist in Tabelle 3.1 zu sehen. In diesem Vergleich wurden die Netzwerk-Emulation-Tools Anhand verschiedener gewichteter Kriterien bewertet, null beschreibt die schlechteste Bewertung zehn ist die beste Bewertung.

Das Tool Cisco VIRL ist nicht mehr erhältlich, und wurde von Cisco Modeling Labs (CML) abgelöst. Aufgrund der in [7] beschriebenen hohen Lizenzkosten von Cisco Modeling Labs (CML), kann dieses Tool auch für dieses Planspiel nicht verwendet werden. Mininet [10] erfüllt den Großteil der in Kapitel 2 gestellten Anforderungen. Jedoch ist es nur für Einzelplätze konzipiert und nicht Multiuserfähig, auch das Einbinden von benutzerdefinierten Containern wird nicht unterstützt. Zusätzlich muss der Aufbau der Infrastruktur zum Start der Emulation vollständig beschrieben sein, eine dynamische Änderung ist nicht möglich. Mit diesen Einschränkungen kann Mininet [10] nicht ohne Modifikationen verwenden werden.

Ein ähnliches Tool zu Mininet [10] ist Kathará [12] [13], wie Mininet basiert es auf Containertechnologien um Systeme zu virtualisieren. Zusätzlich zu den Funktionen die von Mininet bereitgestellt werden, bietet Kathará auch die Möglichkeit benutzerdefinierte Container einzubinden, so können auch Web-Server, DNS-Server oder andere Serverdienste realisiert werden. In [14] werden die Unterschiede einer älteren und einer überarbeiteten Implementation von Kathará beschrieben, beide Implementationen verwenden Docker Con-

		mininet		VIRL		CML		GNS3 (V2.1)	
Kriterium	Gewicht 1-7	Bemerkung	#	Bemerkung	#	Bemerkung	#	Bemerkung	#
Administrativer Aufwand									
- Lizenz (Kosten)	7	Frei (Open Source)	10	199€ für 20 Cisco Nodes p.a. (nicht skalierbar, Einzelplatz-Lizenz)	4	Kostenintensiv (> $2.500 p.a. für 15+10 Nodes), skalierbar (zus. Kosten)	0	Frei (Open Source, GPLv3)	10
- Zentrales Management	3	Isoliert (Einzelplatz), aber per Skript automatisierbar	7	Cluster mit mehreren Compute Nodes (OpenStack)	10	Cluster mit mehreren Compute Nodes (OpenStack)	10	Mehrere aber nicht zentral verwaltete Server möglich	3
- Kompatibilität und Zugänglichkeit	2	verfügbar für Linux (Win, Mac als VM)	7	VMMaestro Client und Server VMs für Linux, Win, Mac, vSphere	8	VMMaestro Client und Server VMs für Linux, Win, Mac, vSphere	8	Verfügbar für Linux, Win, Mac, auch als vorkonfigurierte VM	7
- Custom Images für Netzhardware	4	Kein Support für Images realer Netzhardware	0	Erweiterbar (nur für Cisco Nodes beschränkt), Third-Party Images verfügbar	5	Erweiterbar (nur für Cisco Nodes beschränkt), Third-Party Images verfügbar	5	Images für Vielzahl Router, Switches (Cisco Images nicht enthalten)	7
- Load Balancing	2	Nur manuell	5	Ja	10	Ja	10	Nur manuell	5
- Erforderliche Ressourcen	1	Gering (LXC Container)	10	Device-abhängig	6	Device-abhängig	6	Device-abhängig	8
Didaktische Anforderungen									
- Anbindung an reales Netz	4	Ja	8	Ja	8	Ja	8	Ja (NAT über GUI, Bridge manuell)	6
- Mehrere Benutzer und Sitzungen	5	Nur manuell	5	Nein (offiziell nur single user)	8	Ja	10	Ein Projekt pro User, aber Zugriff mehrerer Clients möglich	6
- Konsolenverbindung	2	Beschränkte LXC Konsole	6	Jeder Benutzer hat eigene Konsole	6	Jeder Benutzer hat eigene Konsole	6	Gemeinsame Konsole (User)	10
- QoS Emulation auf Links	1	Ja	10	Ja (nur Delay, Jitter, Loss)	8	Ja (nur Delay, Jitter, Loss)	8	Ja (nur Delay, Jitter, Loss)	8
- VPN Zugang zu virtuellen Netzen	1	Manuell einrichtbar	5	Ja	10	Ja	10	über VPN Server in virtuellem Netz	5
Gewichtete Bewertung			209		222		204		226

Tabelle 3.1.: „Evaluierung im NetLab eingesetzter Netz-Emulatoren basierend auf Erfahrungswerten“ Quelle: [7]

tainer [15]. In der aktuelle Implementation von Kathará wird zusätzlich ein eigens entwickelter Kubernetes [16] Manager verwendet. Dieser Kubernetes Manager wird in [17] genauer beschrieben. Um die Netzwerkverbindungen zu realisieren, werden in der neu Implementierten Version VXLANs [18] verwendet [17], die ältere Version basiert ausschließlich auf Docker „bridge“ [19] Netzwerken. Dynamische Änderungen der Infrastruktur sind, im Gegensatz zu Mininet, möglich. Jedoch ist Kathará, gleich wie Mininet, nur für Einzeluser konzipiert. Somit kann Kathará nicht ohne größere Modifikationen für das Planspiel verwendet werden.

Da kein Netzwerk-Emulation-Tool verwendet werden kann, und eine Modifikation von Kathará [12] oder Mininet [10] nicht angedacht ist, wird eine eigenständige Lösung implementiert. Die in Kathará und Mininet verwendeten Softwareelemente und Methoden werden dafür als mögliche Basis verwendet.

3.1. Technologien für Virtuelle Maschinen

Bei der Evaluierung einer möglichen Virtualisierungssoftware werden Produkte welche nicht für den Betrieb als Serversoftware konzipiert sind, z.B. VMware Workstation [2] oder VirtualBox [20], ausgeschlossen. Da VMware Workstation bereits zur Realisierung des Planspiels verwendet wurde, und eine weitere Verwendung ähnliche Desktop-Software vermieden wird.
Evaluiert werden folgende ausgewählte Produkte:

- VMware vSphere [21]
- KVM [22]
- Hyper-V [23]
- Proxmox VE [24]

Ausgewählt wurden diese Plattformen, da diese die verbreitetsten Virtualisierungsplattformen sind, in Abbildung 3.1 und [25] wird dies verdeutlicht. Zusätzlich zu den in [25] und [26] angeführten Produkten wurde Proxmox VE [24] ausgewählt, da diese Virtualisierungsplattform bereits in den Netzwerktechnik Laboren der Fachhochschule St.Pölten verwendet wird und praktische Erfahrung mit dieser Plattform vorhanden ist.

Um einen Vergleich der Produkte zu ermöglichen wird folgende Anforderungsliste, welche sich aus den in Kapitel 2 gestellten Anforderungen zusammensetzt, verwendet:

- Container werden unterstützt
- Möglichkeit virtuelle Netzwerke zu erstellen
- Verwaltung der virtuellen Umgebung durch eine Application Programming Interface (API)
- Open Source Software

Abbildung 3.1.: „Magic Quadrant for x86 Server Virtualization Infrastructure“ Quelle: [26]

3.1.1. VMware vSphere

VMware vSphere [21] besteht aus den Produkten VMware ESXI [27] in Kombination mit VMware vCenter [28]. VMware ESXI [27] ist die Hypervisor-Software, um diese zu verwalten wird VMware vCenter [28] benötigt. Auf VMware ESXI [27] Server können per default nur VMs, aber keine Container erstellt und verwaltet werden.

Mit dem zusätzlichen Produkt VMware NSX Data Center [29] können virtuelle Netzwerke in einer VMware vSphere Umgebung erstellt und verwaltet werden. Mithilfe dieser integrierten Software-defined Networking (SDN) Funktionalität können alle in Abschnitt 2.1 gestellten Anforderungen erfüllt werden.

Um Container in einer VMware vSphere [21] Umgebung erstellen und verwalten zu können, sind zwei verschiedene Produkte als mögliche Lösung verfügbar.
„vSphere with Tanzu“ [30] ist ein Produkt, mit dem es möglich ist, Kubernetes [16] Cluster zu erstellen und zu verwalten. Der Fokus dieses Produkts liegt bei Cluster um Entwickler zu unterstützen, nicht bei der Verwaltung einzelner Container um z.B. virtuelle Netzwerke aufzubauen.
Mit dem Open-Source Projekt „vSphere Integrated Containers“ ist es möglich, Docker Container [15] in einer vSphere Umgebung zu erstellen und zu verwalten. Hier ist der Fokus bei der Verwaltung einzelner Container, somit können die Anforderungen von dieser Lösung besser erfüllt werden.

Ein API für VMware vSphere ist verfügbar [31], damit kann die gesamte virtuelle Umgebung verwaltet werden. Auch zusätzliche Funktionalitäten, wie VMware NSX Data Center [29] oder „vSphere with Tanzu“ [30], können über diese API verwaltet werden. Die API kann mithilfe eines, für verschiedene Programmiersprachen erhältlichem, Software Development Kits (SDK) verwendet werden. Die Dokumentation der API ist nach eigenem Empfinden sehr umfassend, aber nicht einfach zu überblicken.

VMware vSphere [21] ist eine Proprietäre Software, für welche Lizenzkosten anfallen. Für erweiterte Funktionalitäten (VMware NSX Data Center [29] oder „vSphere with Tanzu“ [30]) fallen zusätzliche Lizenzkosten an.

Anforderung	Bewertung
Container werden unterstützt	Durch Verwendung eines zusätzlichen Produktes
Möglichkeit virtuelle Netzwerke zu erstellen	Durch Verwendung eines zusätzlichen Produktes
Verwaltung der virtuellen Umgebung durch eine API	Ja
Open Source Software	Nein

3.1.2. KVM

KVM [22] ist ein Linux Kernel Modul, das standardmäßig in jedem Linux Kernel (ab 2.6.20) enthalten ist. Mithilfe der Open Source Software QEMU [32] kann das Kernel Modul verwendet und somit VMs virtua-

lisiert werden.

QEMU/KVM kann standardmäßig nur von der Kommandozeile konfiguriert werden. Dabei ist nur die Verwaltung von VMs vorgesehen. Es gibt einige Management Tools, welche verschiedene Funktionalitäten hinzufügen, eine umfangreiche Liste dieser Management Tools ist in [33] zu finden.

QEMU/KVM hat kein alleinstehendes SDN Modul um virtuelle Netzwerke zu verwalten, aber es kann in Abschnitt 3.2 beschriebene Software in Kombination mit QEMU/KVM verwendet werden, um SDN Funktionalitäten zu realisieren.

QEMU/KVM bietet eine minimale Oberfläche, um Virtuelle Maschinen zu verwalten, die in Kapitel 2 beschriebenen Anforderungen werden nicht erfüllt. In Kombination mit in [33] beschriebenen Management Tools und in Abschnitt 3.2 beschriebener Software zur Verwaltung virtueller Netzwerke kann QEMU/KVM als mögliche Lösung verwendet werden.

Anforderung	Bewertung
Container werden unterstützt	Mithilfe zusätzlicher Software
Möglichkeit virtuelle Netzwerke zu erstellen	Mithilfe zusätzlicher Software
Verwaltung der virtuellen Umgebung durch eine API	Mithilfe zusätzlicher Software
Open Source Software	Ja

3.1.3. Hyper-V

Hyper-V ist ein Hypervisor, welcher in Windows Server [34] und Windows 10 [35] integriert ist. Mithilfe von Hyper-V können VMs erstellt und verwaltet werden.

Hyper-V stellt Funktionen zur Erstellung von virtuellen Switches zur Verfügung [36]. Auch Funktionen zur Verwaltung von virtuellen Netzwerken werden angeboten, diese stehen jedoch nur für Windows Server [34] zur Verfügung [37].

Hyper-V/Windows unterstützt das Erstellen und Verwalten von Containern, verwendet werden dabei Docker Container [15]. Zusätzlich wird Kubernetes [16] unterstützt. Hyper-V/Windows kann Docker Container [15] basierend auf Windows und Linux verwalten. Dabei können nicht nur bereitgestellte Images, sondern auch

angepasste Images verwendet werden. [38]

Hyper-V stellt verschiedene APIs [39] für die unterschiedlichen Funktionalitäten zur Verfügung (VMs, Netzwerke, Container). Die Dokumentation der APIs ist, ähnlich zu VMware vSphere [21], nach eigenem Empfinden sehr umfassend aber nicht einfach zu überblicken.

Um alle nötigen Funktionalitäten zu erhalten, muss Windows Server [34] verwendet werden. Für diese Software fallen ebenfalls Lizenzkosten an.

Anforderung	Bewertung
Container werden unterstützt	Ja
Möglichkeit virtuelle Netzwerke zu erstellen	Ja, mehr Funktionalitäten nur bei Verwendung von Windows Server
Verwaltung der virtuellen Umgebung durch eine API	Ja
Open Source Software	Nein

3.1.4. Proxmox VE

Proxmox VE [24] ist eine Virtualisierungsplattform basierend auf dem Debian [40] Linux Derivat. Proxmox VE ist eine Open Source Plattform bei der ist auch möglich ist ein zusätzliches Support Abonnement abzuschließen. Diese Zusatzleistung gibt Zugang zu einem stabilen „Enterprise Repository“ und verschiedene Support-Leistungen der Entwickler. [41]

Zur Verwendung von VMs wird der in Unterabschnitt 3.1.2 beschriebene Hypervisor KVM verwendet. Proxmox VE wird auch in der Liste der Management Tools [33] für KVM aufgelistet.
LXC Container [42] können standardmäßig auf der Plattform Proxmox VE erstellt und verwaltet werden. Keine zusätzlichen Installationen von Software sind dafür notwendig.

Proxmox VE unterstützt Netzwerk „bridges“, diese sind eine simple Implementierung eines virtuellen Switches. Zusätzlich zu dieser Funktionalität können komplexere virtuelle Netzwerke, mithilfe der von Linux zur Verfügung gestellten Netzwerkfunktionen, erstellt werden. Einige dieser Funktionen sind VLan, VXLan, VRF und virtuellen Netzwerkbrücken [43] [44].
Ein SDN Modul [45] für Proxmox VE ist in Entwicklung, wird allerdings noch als „experimental feature“ gekennzeichnet. Mithilfe dieses Moduls können komplexe Virtuelle Netzwerke auf der Virtualisierungs-

plattform erstellt und verwaltet werden.

Eine sehr gut dokumentierte und einfach zu verstehende API ist vorhanden [46]. Mit deren Hilfe können alle Funktionen der Virtualisierungsplattform gesteuert und verwaltet werden. Auch das als „experimental feature“ gekennzeichnete SDN Modul kann mithilfe der API verwaltet werden.

Die standardmäßige Integration von LXC Containern [42], das noch in Entwicklung befindliche SDN Modul und die umfangreiche API erfüllen die gegebenen Anforderungen. Wird keine zusätzliche Support-Leistung der Proxmox VE Entwickler benötigt, fallen keine Lizenzkosten an.

Anforderung	Bewertung
Container werden unterstützt	Ja
Möglichkeit virtuelle Netzwerke zu erstellen	Ja, noch in Entwicklungsphase
Verwaltung der virtuellen Umgebung durch eine API	Ja
Open Source Software	Ja

3.1.5. Container Technologien

Wie bereits durch die Umsetzung von Mininet [10], Kathará [12] und dem Paper [9] verdeutlicht wird, sind Container wesentlich ressourcenschonender und performanter als VMs. In [47] und [48] wird diese Vermutung bestätigt. Deswegen wird eine Verwendung von Containern als Server und Router angestrebt. In diesem Abschnitt werden die beiden Container Lösungen Docker [15] und LXC [42] evaluiert.

Bei beiden Lösungen werden so genannte Images verwendet um Container zu erstellen. Basis Images werden von den jeweiligen Produkten mitgeliefert. Diese Basis Images können modifiziert werden, so können Container mit bereits vorgefertigter Konfiguration erstellt werden.

LXC [42] verwendet einige Linux Kernel Funktionen um Prozesse voneinander zu trennen, somit sind alle LXC Container nur Prozesse die voneinander und vom Host-System virtuell getrennt sind. Gleichzeitig teilen sich alle LXC Container den selben Kernel vom Host-System, wie in Abbildung 3.2 veranschaulicht wird. So werden verwendeten Ressourcen möglichst gering gehalten.
Das Konzept von Docker [15] basiert auf dem selben wie LXC, auch hier werden Linux Kernel Funktionen verwendet um Prozesse voneinander zu trennen. Anders als LXC verwendet Docker eine Engine, welche

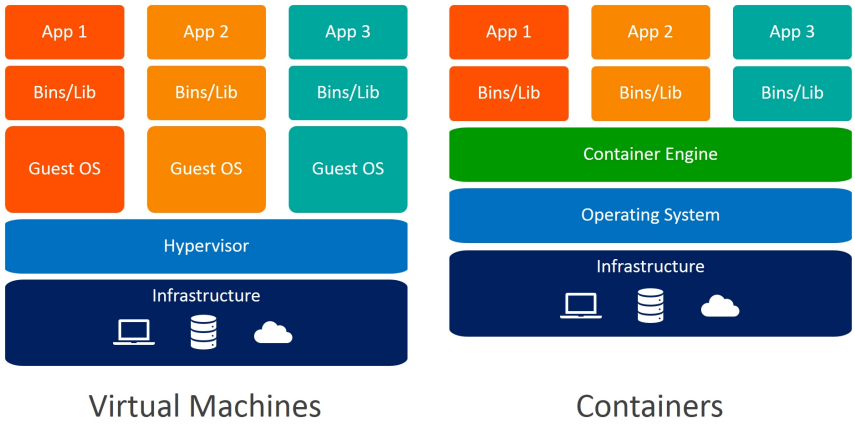


Abbildung 3.2.: Vergleich zwischen VMs und Containern Quelle: [49]

über eine API verwendet werden kann. [50] Diese Docker Engine bietet mehr Funktionalitäten als LXC.

LXC [42] und Docker [15] sind Open-Source Software, wobei Docker auch verschiedene Produkte anbietet welche Lizenzkosten benötigen. [51]

Aus [52] geht hervor, dass LXC Container [42] besser für das virtualisieren von Linux Server sind, Docker Container [15] sind eher für die Virtualisierung von verschiedenen Applikationen gedacht. Beide Lösungen eignen sich für den Anwendungsfall dieser Arbeit, LXC Container werden jedoch bevorzugt da gesamte Systeme und keine einzelnen Applikationen virtualisiert werden sollen.

3.2. Virtuelle Netzwerke

SDN ist kein Produkt an sich, aber ein Begriff der ein Konzept beschreibt. Dieses Konzept beschreibt programmierbare und automatisch veränderbare Netzwerke. [53]

Die Definition von SDN laut der Open Networking Foundation (ONF) ist:
„In the SDN architecture, the control and data planes are decoupled, network intelligence and state are logically centralized, and the underlying network infrastructure is abstracted from the applications.“ [54]

Ein weit verbreitetes Protokoll, welches zur Implementation von SDN verwendet wird ist OpenFlow. OpenFlow ermöglicht es die Entscheidungen zur Weiterleitung von Netzwerkpaketen von Router und Switches zu kontrollieren. So können Router und Switches verschiedener Anbieter, mithilfe eines Controllers, über

ein einheitliches Protokoll aus der Ferne verwaltet werden. [55] [56]

Wie in Unterabschnitt 3.1.1 und Unterabschnitt 3.1.3 beschrieben, implementieren einige Hypervisor Lösungen eigene virtuelle Switches. Eine Open Source Implementierung für virtuelle Switches ist Open vSwitch [57]. Open vSwitch unterstützt unter anderem das OpenFlow Protokoll [56]. Open vSwitch kann auf einigen Hypervisor Plattformen und mit Container Lösungen eingesetzt werden. [58]

Virtuelle Netzwerke müssen nicht mit einem virtuellen Switch oder einem SDN Controller verwaltet werden, eine einfache Möglichkeit virtuelle Netzwerke zu erschaffen und zu verwalten ist VXLAN [18]. Diese Technologie kann virtuelle Layer-2 Netzwerke auf Layer-3 Overlay-Netzwerken transportieren. [18] VXLAN ist im Linux Kernel implementiert, und kann so standardmäßig von jedem Linux System verwendet werden. [43]

3.3. Routing Software

Da eine Verwendung von Containern zur Virtualisierung von Routern angestrebt wird, muss Routing Software die auf diesen Containern verwendet werden kann evaluiert werden. Cisco IOS steht nur als VM zur Verfügung, als Beispiel dient der Virtuelle Router Cisco CSR 1000V [5]. Die Routing Software soll Ähnlichkeiten zu dem Betriebssystem Cisco IOS haben, um den Studierenden die Anwendung zu erleichtern, da diese im Zuge des Studiums Erfahrungen mit Cisco IOS gesammelt haben. Die Routing Protokolle BGP und OSPF muss die Software in Verwendung mit IPv6 unterstützen.

Die verschiedenen Routing Softwares werden anhand folgender Anforderungsliste bewertet:

- Auf Containern ausführbar
- Ähnlichkeiten zu Cisco IOS
- Unterstützt BGP und OSPF
- Open Source Software

3.3.1. VyOS

VyOS [59] ist ein Open Source Projekt welches eine Router Plattform anbietet, diese Plattform ist standardmäßig nur als VM erhältlich. Jedoch gibt es Ansätze VyOS auch auf Docker Containern [15] zur Verfügung zu stellen wie [60] und [61] zeigen.

VyOS unterstützt die Routing Protokolle BGP (IPv4 und IPv6), OSPF (IPv4 und IPv6), RIP und RIPng, sowie einige weitere Funktionen wie VPN, IPSEC, VXLAN und Firewalling. [59] Minimale Systemanforderungen sind 500MiB RAM und 2 GiB Festplattenspeicher. [62]

Seit Version 1.2 ist das ISO von VyOS nicht mehr frei zur Verfügung stehend und muss selbst vom Source Code gebaut werden:
„Starting with VyOS 1.2 the release model of VyOS has changed. VyOS is now free as in speech, but not as in beer. This means that while VyOS is still an open source project, the release ISOs are no longer free and can only be obtained via subscription, or by contributing to the community. The source code remains public and an ISO can be built using the process outlined here.“ [63]

Anforderung	Bewertung
Auf Containern ausführbar	Nein, nur durch Modifikationen
Ähnlichkeiten zu Cisco IOS	Ja
Unterstützt BGP und OSPF	Ja
Open Source Software	Ja, muss selbständig vom Source Code gebaut werden

3.3.2. Quagga

Quagga [64] ist ein Open Source Routing Paket für Unix Systeme. Die zur Verfügung gestellte Benutzerschnittstelle hat starke Ähnlichkeit zu der von CISCO IOS [6]. Folgende Routing Protokolle werden unterstützt:

- OSPF
- OSPFv3
- RIP
- RIPNG
- ISIS
- ISISv6
- BGP

Quagga wird seit 2018 nicht mehr aktiv weiterentwickelt. [65]

Anforderung	Bewertung
Auf Containern ausführbar	Ja
Ähnlichkeiten zu Cisco IOS	Ja
Unterstützt BGP und OSPF	Ja
Open Source Software	Ja

3.3.3. FRRouting

FRRouting [66] ist ein Open Source Routing Paket für Linux. Dieses Softwarepaket wird als Debian Paket, als RPM Paket oder als Snap in den jeweiligen Repositorys zur Verfügung gestellt. Da es sich bei FRRouting um ein Softwarepakte handelt, ist die Verwendung dieser in einem Container ohne Probleme durchführbar. Die Benutzerschnittstelle hat, wie Quagga [64], starke Ähnlichkeit zu der von CISCO IOS [6]. FRRouting ist ein Fork von Quagga. [66]

Spezielle Systemanforderungen werden nicht gestellt, jedoch kann das Softwarepaket mit sehr niedrigen Systemanforderungen betrieben werden.
„FRR can be run on very low resource systems such as SBCs, provided it is not stressed too much. If you want to set up 4 Raspberry Pis to play with BGP or OSPF, it should work fine.“ [67]

FRRouting unterstützt einige Routing Protokolle unter anderem folgende:

- OSPF
- OSPFv3
- RIP
- RIPNG
- ISIS
- ISISv6
- BGP

Anforderung	Bewertung
Auf Containern ausführbar	Ja
Ähnlichkeiten zu Cisco IOS	Ja
Unterstützt BGP und OSPF	Ja
Open Source Software	Ja

4. Herangehensweise

Die zur Implementierung verwendeten Technologien wurden aufgrund der in Kapitel 2 beschriebenen Anforderungen ausgewählt.

4.1. Verwendete Technologien

Um die Entwicklung einer Schnittstelle zwischen Spieloberfläche und der virtualisierten Infrastruktur möglichst einfach zu gestalten wurde die in Unterabschnitt 3.1.4 vorgestellte Virtualisierungsplattform Proxmox VE [24] in der Version 6.3-3 verwendet. Zusätzlich wurde das noch in der BETA Phase befindliche SDN Modul [45] verwendet.

Diese Virtualisierungsplattform wurde gewählt, da alle benötigten Anforderungen auf dieser Plattform erfüllt werden können. Eine direkte Virtualisierung von LXC Containern [42] und VMs ohne zusätzliche Module oder Software ist möglich. Auf anderen Virtualisierungsumgebungen ist das nur zusätzliche Module oder der Verwendung von eigens dafür konzipierter Software möglich. Eine Nutzung von zusätzlichen Modulen oder zusätzlicher Software erhöht die Komplexität der zu implementierenden Lösung, und wird daher vermieden. Zusätzlich wird so eine Erhöhung der Lizenzkosten vermieden. Plattformen wie Mininet [10] und Kathará [12] sind bestens zur Emulation von Netzwerken geeignet, jedoch nur zur Nutzung einzelner Personen gedacht, und können daher nicht verwendet werden. Durch die vorhandene Implementierung von SDN und die Möglichkeit diese Funktionen über die API zu steuern verringert sich die Komplexität der zu implementierenden Lösung weiter. Zusätzlich hat die Verwendung unter Open Source Lizenz und die sehr gut Dokumentierte und leicht verständliche API [46] überzeugt.

Um jeweils Server und Router darzustellen wurden dazu LXC Templates erstellt, als Basis Image dient für diese jeweils ein Ubuntu 20.04 [68] LXC Image. Für Router wurde auf diesem Basis Image das in Unterabschnitt 3.3.3 vorgestellte Softwarepaket FRR [66] installiert und entsprechend Konfiguriert, mit dessen Hilfe ist es den Studierenden möglich das geforderte Netzwerk aufzubauen.

Auf das Template für Server wurden folgende Softwarepakete installiert um den Studierenden Möglichkeiten zur Realisierung der Infrastruktur und ihrer erforderlichen Komponenten und zur Fehlersuche zu geben.

- iproute2
- openssh-server
- supervisor
- apache2
- postfix
- rsyslog
- bind9
- bind9utils
- bind9-doc
- traceroute
- iputils-ping
- vim
- nano
- php
- wget
- curl
- tcpdump
- dnsutils
- netcat-openbsd
- telnet

Diese Liste an Softwarepaketen wurde aus eigener Auswahl und auch durch Wünsche von Studierenden, welche in den Durchführungen des Planspiels eingingen, erstellt. Diese Softwarepakete sind bereits vorinstalliert, da ein Zugriff auf das „reale“ Internet aus dem Netzwerk des Planspiels nicht möglich ist. Somit ist das nachinstallieren von Software nicht möglich.

Testweise wurde auch ein Template für eine VM erstellt, und das automatische erstellen von VMs realisiert. Diese VM kann auch von Gruppen erstellt werde, hat jedoch noch keine Funktionalität im Spielverlauf. Das Erstellen von VMs wurde getestet um sicherzustellen, dass auch Applikationen mit Graphical User Interface (GUI) von Gruppen genutzt werden können. Das Ausführen von Applikationen mit Graphical User Interface (GUI) ist auch auf LXC Containern möglich, jedoch ist das nur mithilfe von Remote-Desktop-Anwendungen

umsetzbar. Um Remote-Desktop-Anwendungen zu verwenden muss eine Verbindung in das Netzwerk des Containers möglich sein, das wird jedoch durch den Aufbau des Planspiels unmöglich. So können keine Applikationen mit Graphical User Interface (GUI) auf den verwendeten LXC Containern benutzt werden.

4.2. Technische Umsetzung

Der größte Teil dieser Arbeit war die Integration der virtualisierten Infrastruktur in die bereits bestehende Spieloberfläche. Da die Spieloberfläche mit PHP [69] und JavaScript [70] entwickelt wurde, wurden auch diese Sprachen weiterhin verwendet. Die größten Eingriffe in die Spieloberfläche wurden im Zuge der Integration an den Benutzeroberflächen zur Konfiguration und zum Kauf von Komponenten vorgenommen. Es wurde auch die Durchführung von Ticks verändert, hier wird nun automatisch der Ausfall von Komponenten simuliert. Außerdem wurde die nötige Funktionalität um Komponenten zu erstellen, zu löschen, zu konfigurieren und den Ausfall dieser zu simulieren implementiert. Um den Gruppen Zugang zur Infrastruktur zu ermöglichen wurde auch Funktionalität implementiert um Benutzer und Berechtigungen auf der Virtualisierungsplattform (Proxmox) zu verwalten.

Im Zuge der Integration der virtualisierten Infrastruktur in die Spieloberfläche mussten große Teile der Spieloberfläche neu implementiert werden. Die durch die Spieloberfläche verwendete Datenbank musste angepasst werden, um Informationen zur virtualisierten Infrastruktur verwalten zu können. Da eine genaue Beschreibung aller Änderungen nicht zielführend für diese Arbeit ist, werden nur die verwendeten Aufrufe der Proxmox API beschrieben. Alle Änderungen sind in dem verwendeten (privaten) Gitlab Repository zu finden, welches mit Stand des Abgabedatums auch auf der beiliegenden CD vorzufinden ist.

In Abbildung 4.1 wird der standardmäßige Ablauf zu Änderungen der virtuellen Infrastruktur dargestellt. Zuerst werden Daten, zu den Änderungen, von den Studierenden auf der Spieloberfläche angegeben, wie in Unterabschnitt 4.2.1 beschrieben. Diese Daten werden auf Validität geprüft, etwa wird das Format der IPv6 Adressen geprüft. Auch wird geprüft ob eine Gruppe die nötigen Berechtigungen besitzt um die angegebenen Änderungen durchzuführen, Gruppen sind nur dazu in der Lage ihre eigenen Komponenten zu ändern. Zusätzlich werden Einschränkungen durch Spielmechaniken und Spielregeln angewandt. Nach erfolgreicher Validierung der angegebenen Daten werden die für die Änderungen notwendigen API Aufrufe abgesetzt. Schlägt ein API Aufruf fehl, werden alle bereits durchgeführten Aufrufe zurückgesetzt. Bei erfolgreicher Umsetzung der Änderungen werden die Änderungen in der Datenbank der Spieloberfläche gespeichert und

der Erfolg auf der Spieloberfläche angezeigt. Bei einem Fehlschlag wird eine entsprechende Fehlermeldung dargestellt.

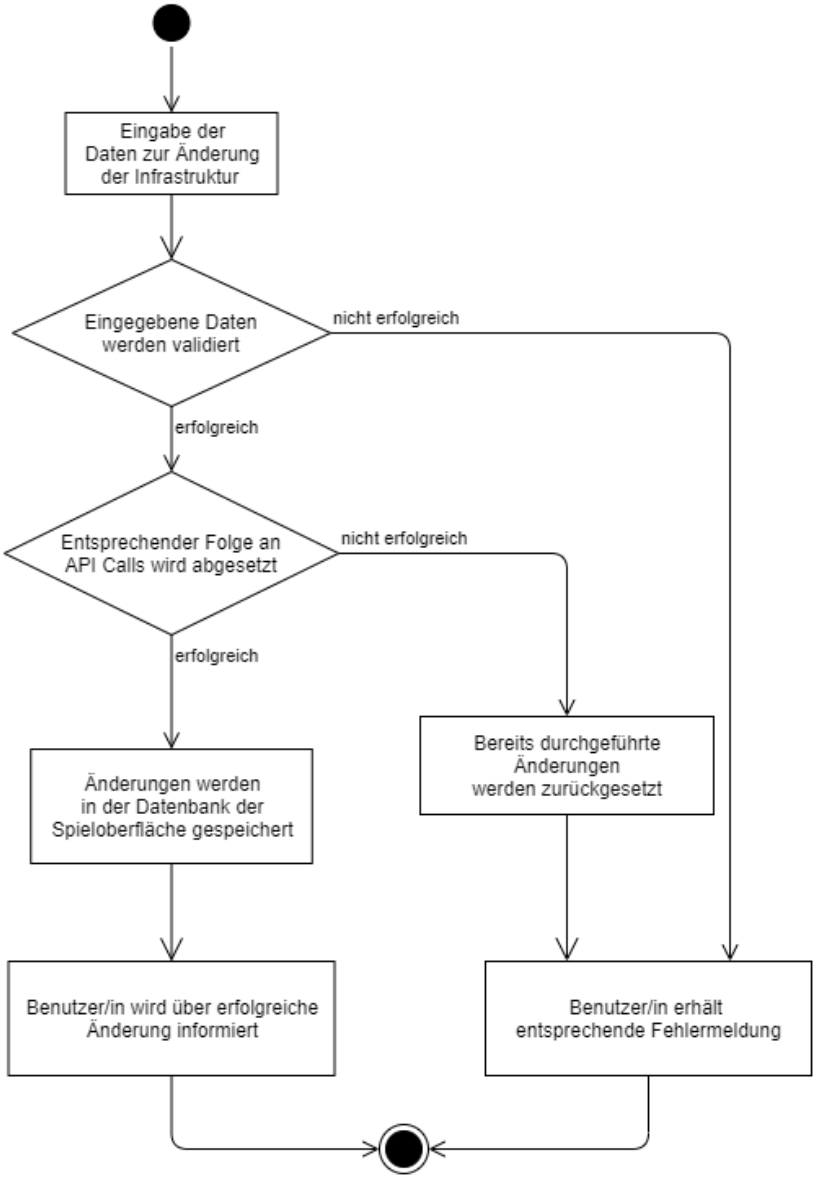


Abbildung 4.1.: Standardmäßiger Ablauf einer Änderung der virtuellen Infrastruktur

Folgende Aufrufe der Proxmox API werden in der Implementation verwendet:

- nodes/node/tasks/upid/status

  Mit dieser API Funktion kann der Status eines Tasks (z.B. die Erstellung eines LXC Containers) überprüft werden

- nodes/node/lxc

  Wird die HTTP GET Methode verwendet, werden alle LXC Container gezeigt

  Bei Verwendung der POST Methode kann ein neuer LXC Container erstellt werden.
- nodes/node/qemu

  Zeigt alle Virtuelle Maschinen (VMs)
- access/users

  Hier wird die HTTP POST Methode verwendet um einen neuen Benutzer auf der Virtualisierungsplattform zu erstellen.
- access/users/userid

  Dieser API Pfad wird mit der HTTP DELETE Methode verwendet, um einen Benutzer zu löschen

  Auch wird dieser Pfad mit der HTTP PUT Methode verwendet um Benutzer zu deaktivieren/aktivieren.
- cluster/nextid

  Diese Funktion wird benötigt, um eine freie ID für eine neue VM oder einen neuen LXC Container zu finden.
- nodes/node/qemu/vmid/clone

  Die HTTP POST Methode wird verwendet um eine VM oder ein Template zu klonen.
- nodes/node/lxc/qemu/vmid/firewall/options

  Hier kann die HTTP PUT Methode verwendet werden um Firewall Optionen einer VM/eines Containers zu ändern.
- nodes/node/lxc/qemu/vmid/firewall/rules

  Die HTTP POST Methode wird verwendet, um eine neue Firewall Regel an dieser VM/diesem Container hinzuzufügen.
- access/acl

  Die HTTP PUT Methode wird verwendet, um Berechtigungen der Benutzer der Virtualisierungsplattform zu verwalten.
- cluster/sdn/zones

  Um alle verfügbaren Netzwerkzonen anzuzeigen wird die HTTP GET Methode verwendet

  Die HTTP POST Methode wird verwendet, um eine neue Netzwerkzone zu erstellen.
- cluster/sdn/vnets

  Die HTTP POST Methode wird verwendet, um ein neues Virtuelles Netzwerk zu erstellen.
- cluster/sdn

Hier wird die HTTP PUT Methode verwendet, um alle Änderungen im SDN anzuwenden.

- nodes/node/lxc/qemu/vmid/config

  Die HTTP PUT Methode wird verwendet, um Einstellungen einer VM/eines Containers zu verändern.
- nodes/node/lxc/qemu/vmid/status/start

  Die HTTP POST Methode wird verwendet, um eine VM/einen Container zu starten.
- nodes/node/lxc/qemu/vmid/status/shutdown

  Die HTTP POST Methode wird verwendet, um eine VM/einen Container zu stoppen.
- nodes/node/lxc/qemu/vmid

  Die HTTP DELETE Methode wird verwendet, um eine VM/einen Container zu löschen.
- cluster/sdn/vnets/vnet

  Die HTTP DELETE Methode wird verwendet, um ein Virtuelles Netzwerk zu löschen.
- cluster/sdn/zones/zone

  Die HTTP DELETE Methode wird verwendet, um eine Netzwerkzone zu löschen.

4.2.1. Integration in die Spieloberfläche

Um die Konfiguration der virtuellen Infrastruktur so einfach als möglich zu gestalten, wurde versucht, den Kauf und Konfigurationsprozess in die Spieloberfläche einzubinden.

Kauf von Komponenten

Bei dem Kauf eines Servers oder Routers muss, wie in Abbildung 4.2 zu sehen ist, ein Hostname, ein Passwort und das Land in dem sich die Komponente befinden soll, von der Gruppe angegeben werden. Der Hostname dient zur leichteren Identifikation des Containers in der Virtualisierungsumgebung und wird auch automatisch als „hostname“ auf dem jeweiligen Container konfiguriert. Das Passwort, um sich am Server bzw. Router anzumelden muss ebenfalls angegeben werden und wird auch automatisch auf dem LXC Container konfiguriert. Die angegebenen Passwörter werden nicht gespeichert.
Wie in Abbildung 4.3 zu sehen ist, ist der Kauf eines „Desktop PCs“ ähnlich, jedoch muss hier kein Passwort angegeben werden. Eine automatische Konfiguration eines Passworts ist nur auf LXC Containern möglich, da der „Desktop PC“ als VM realisiert wurde, sind diese mit einem Standard Passwort konfiguriert, das allen Gruppen mitgeteilt wird. Die Gruppen müssen selbständig die Passwörter ändern, um übernahmen von anderen Gruppen zu verhindern.
Das Benutzerinterface zum Kauf einer Netzwerkverbindung wurde nicht verändert, hier können die Länder angegeben werden, welche verbunden werden. Dies wird in Abbildung 4.4 veranschaulicht.

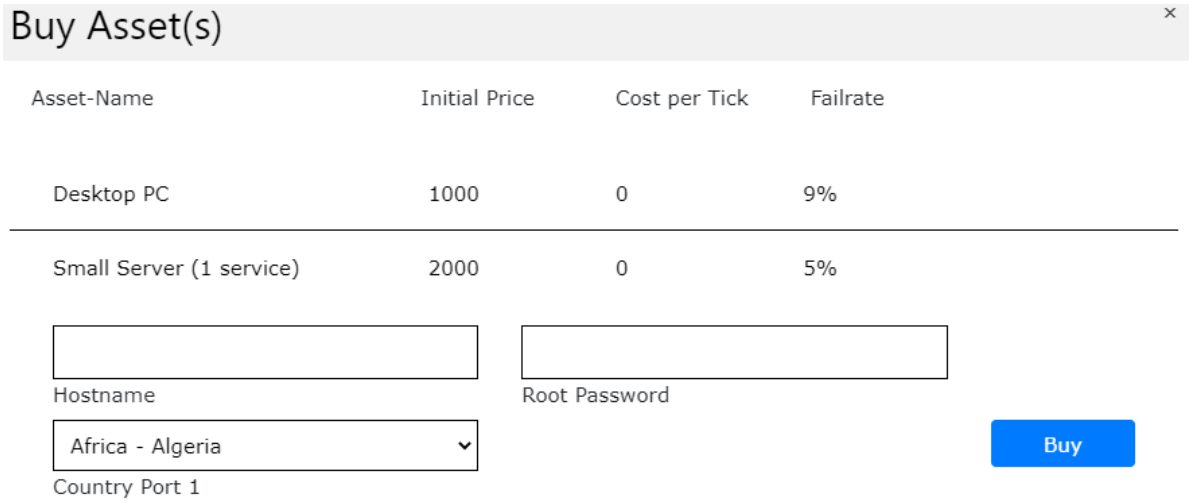


Abbildung 4.2.: Menü zum Kauf eines Servers/Routers

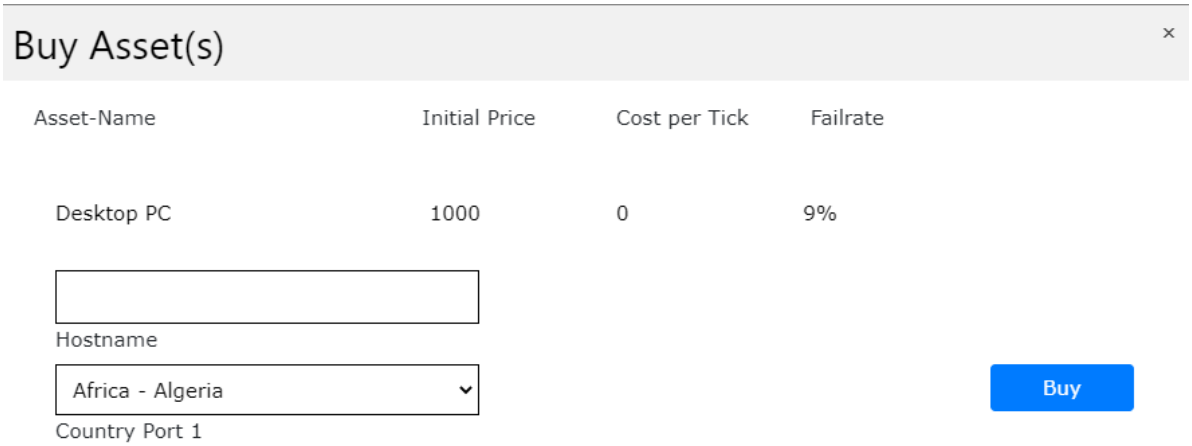


Abbildung 4.3.: Menü zum Kauf eines Desktop PCs

Konfiguration von Komponenten

Um in das Konfigurationsmenü einer Komponente zu gelangen, können die in Abbildung 4.5 rot markierten Schaltflächen verwendet werden. In diesem Menü werden alle Konfigurationsoptionen gezeigt.

Um einen Server oder einen Router mit dem gewünschten Netzwerk zu verbinden, kann dieses ausgewählt werden. Wie in Abbildung 4.6 zu sehen ist, werden nur Netzwerkverbindungen angezeigt die anhand der Spielelogik auf für diesen Server oder Router möglich sind (befinden sich im selben Land, die Gruppe hat die Berechtigung einer anderen Gruppe erhalten diese Netzwerkverbindung zu verwenden).

Nachdem die gewünschte Netzwerkverbindung gewählt wurde, kann eine IPv6 Adresse eingetragen werden.

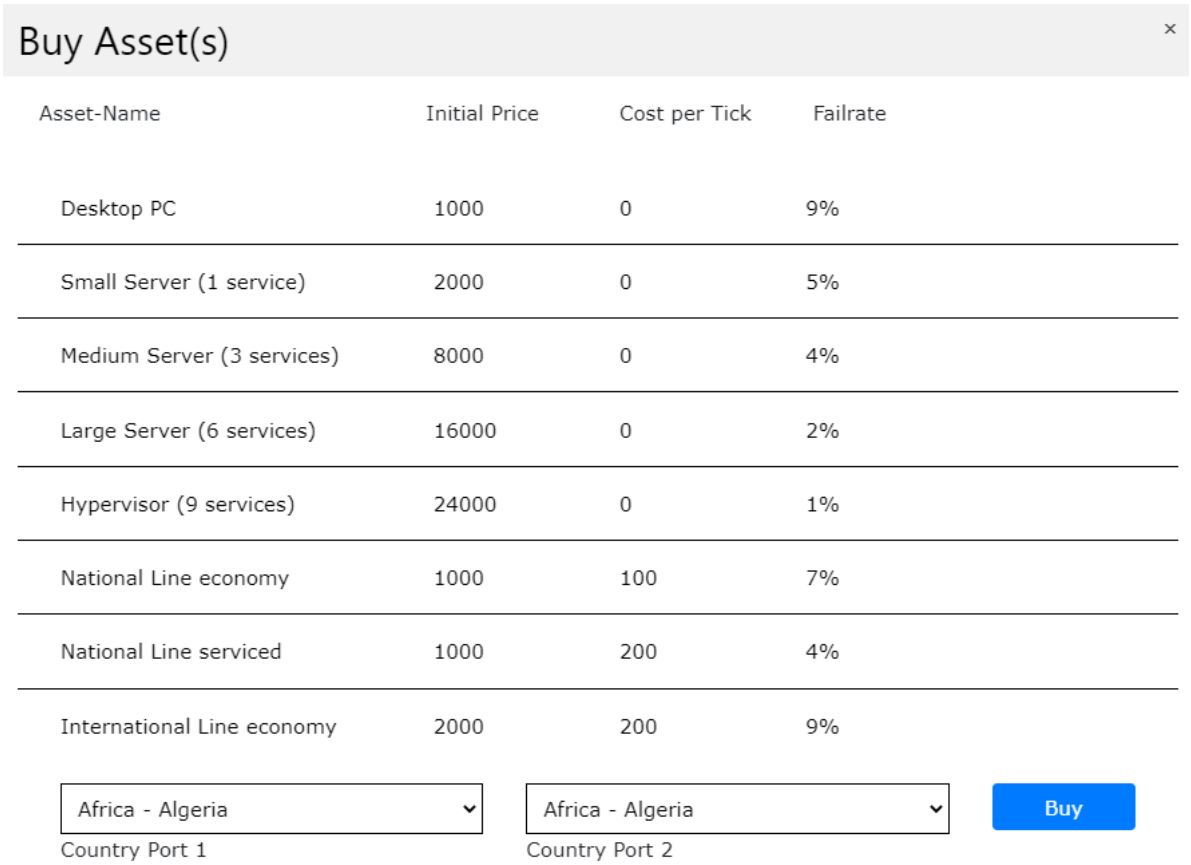


Abbildung 4.4.: Menü zum Kauf einer Netzwerkverbindung

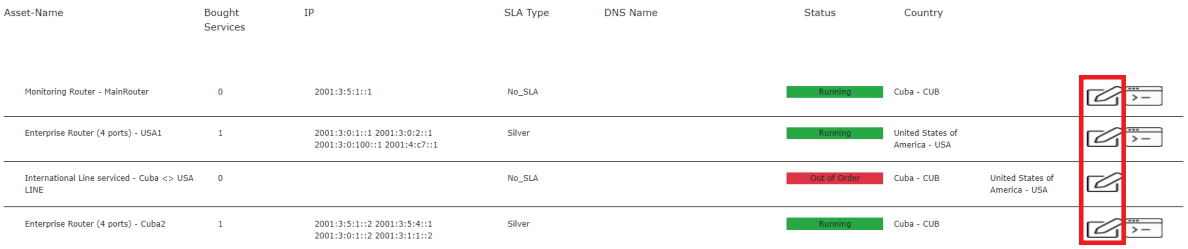


Abbildung 4.5.: Darstellung einzelner Komponenten in der Spieloberfläche

Nachdem die Konfiguration gespeichert wird, wird der Router bzw. Server mit dem jeweiligen Netzwerk automatisch verbunden, und die eingetragene IPv6 Adresse wird für das gewählte Interface konfiguriert. Eine solche Konfiguration ist in Abbildung 4.7 dargestellt.

4.2.2. Zugriff der Spieler/innen auf die Infrastruktur

Vor Beginn eines neuen Planspiels müssen von den Administratoren/Administratorinnen die benötigten Gruppen Logins in der Spieloberfläche angelegt werden. Beim Anlegen der Gruppen wird nun zusätzlich

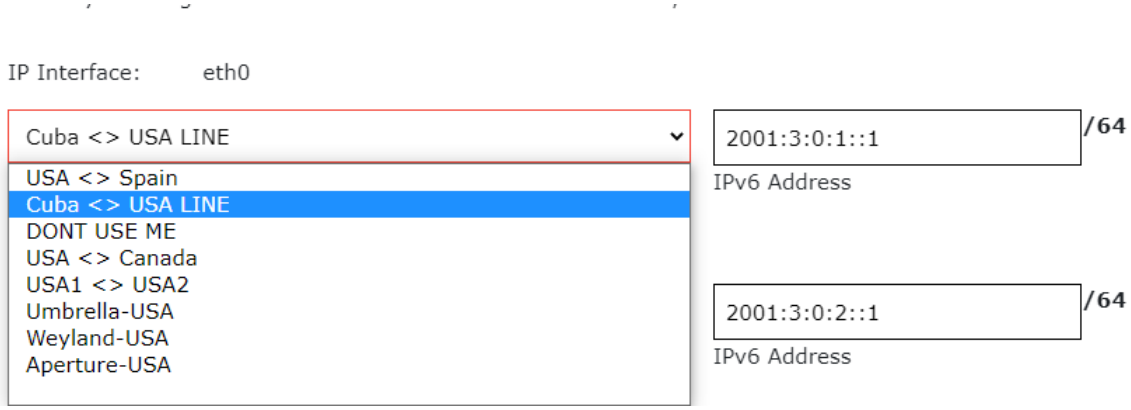


Abbildung 4.6.: Auswahl der Netzwerkverbindung bei der Konfiguration

ein Benutzer pro Gruppe auf der Virtualisierungsplattform angelegt. So ist es den Gruppen möglich, sich auf der Virtualisierungsplattform einzuloggen und auf ihre Komponenten zuzugreifen.

Wird das Planspiel z.B. am Ende eines Tages pausiert, werden alle Benutzer der Gruppen auf der Virtualisierungsplattform deaktiviert. Auch der Login auf die Spieloberfläche ist in dieser Zeit nicht möglich. Diese Maßnahme wurde ergriffen um unfaire Vorteile, für Gruppen die nach Ende der regulären Spielzeit weiterarbeiten, zu verhindern. Bei Fortsetzung des Planspiels werden die Benutzer wieder aktiviert, und der Zugriff auf die Virtualisierungsplattform und die Spieloberfläche ist den Gruppen wieder möglich. In Abbildung 4.8 ist die Benutzerverwaltung, während einer Spielpause, auf der Virtualisierungsplattform abgebildet.

Um von der Spieloberfläche zur Virtualisierungsplattform zu gelangen, können die in Abbildung 4.9 rot markierten Schaltflächen verwendet werden.

Die Berechtigungen der Gruppen auf der Virtualisierungsplattform sind sehr stark eingeschränkt. Den Gruppen ist es nur möglich ihre eigenen Komponenten zu sehen und auf die Console dieser zuzugreifen. Dadurch wird verhindert, dass Komponenten anderer Gruppen manipuliert werden. Einstellungen der Komponenten können angezeigt, jedoch nicht verändert werden, da ein Verändern der Einstellungen das Umgehen von einigen Spielmechaniken (z.B. dem Ausfall von Komponenten) ermöglichen würde. Das Erstellen von neuen Komponenten ist den Gruppen nur über die Spieloberfläche möglich, in der Virtualisierungsplattform ist das nicht möglich. Diese Einschränkung wird durch die in Abbildung 4.10 gezeigten grauen Schaltflächen, zum Starten/Stoppen einer Komponente und zur Erstellung einer neuen VM/eines neuen Containers, dargestellt.

In Abbildung 4.11a und Abbildung 4.11b wird die unterschiedliche Darstellung der Komponenten von zwei verschiedenen Gruppen gezeigt.

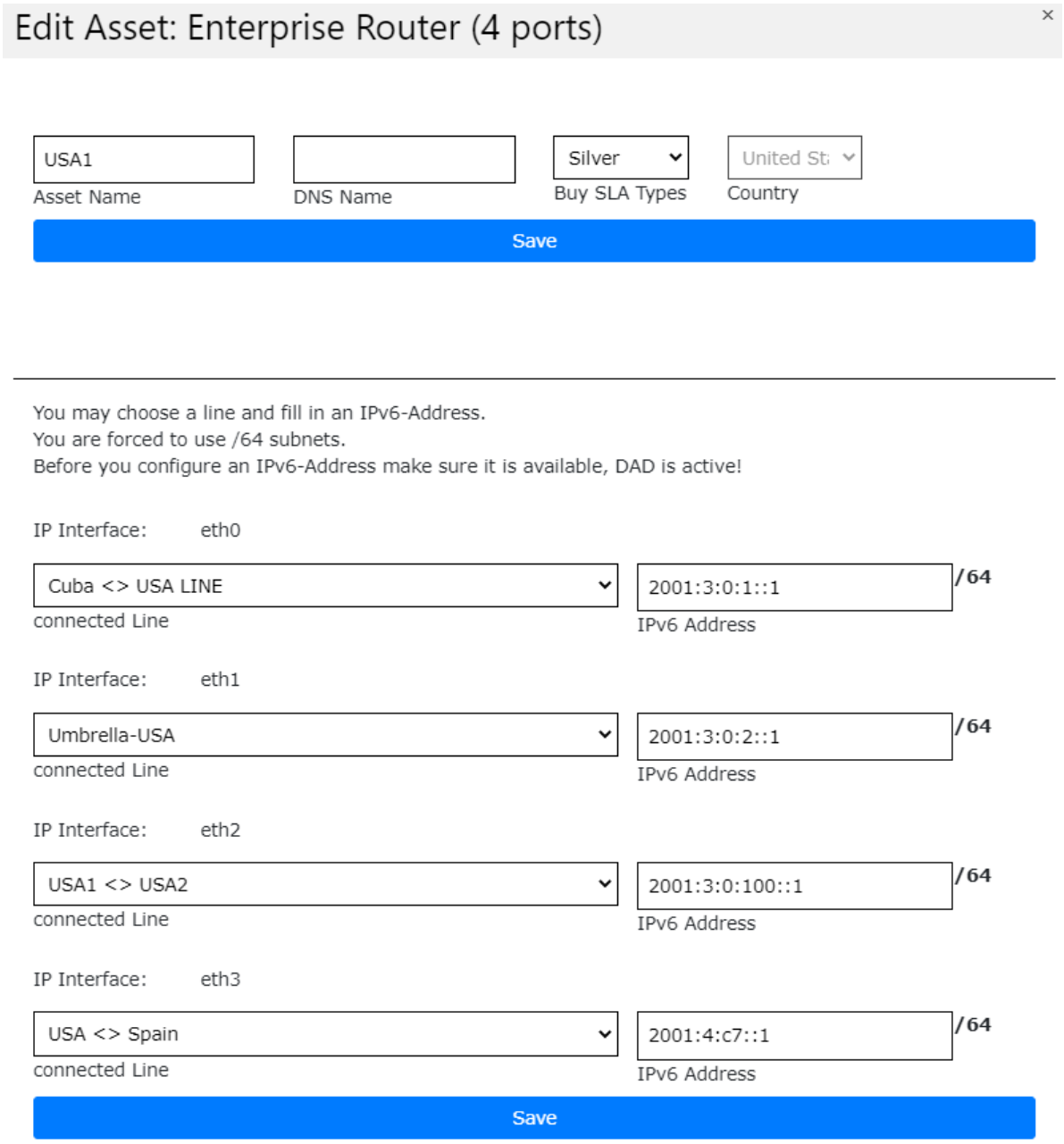


Abbildung 4.7.: Menü zum Konfigurieren eines Servers/Routers

Um auf eine Komponente zuzugreifen und diese zu Konfigurieren kann der „Console“-Tab in der Virtualisierungsumgebung benutzt werden (Abbildung 4.12).

Die in der Spieloberfläche vorgenommene Netzwerkkonfiguration kann von den Gruppen auf der Virtualisierungsplattform überprüft werden, wie in Abbildung 4.13 gezeigt wird.

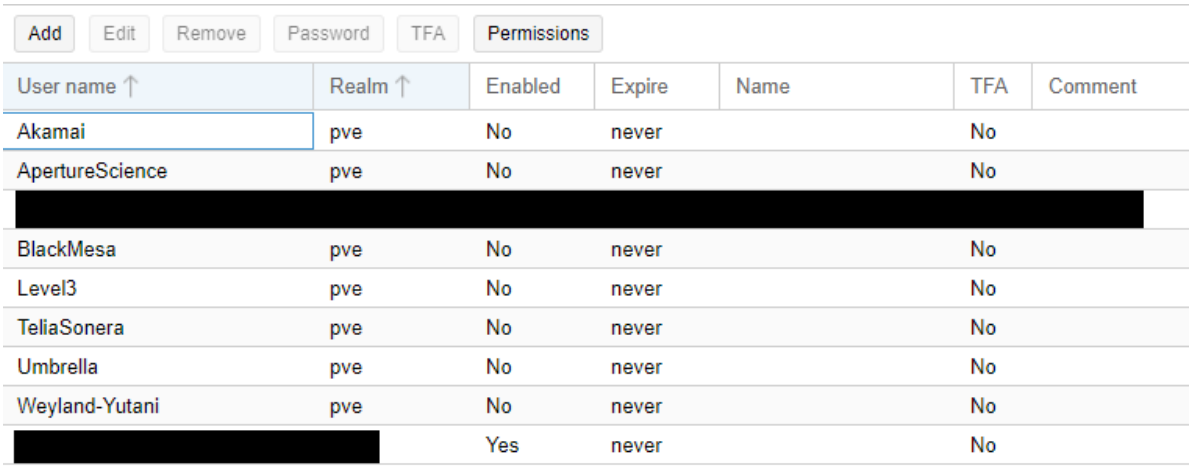


Abbildung 4.8.: Benutzerverwaltung auf der Virtualisierungsplattform

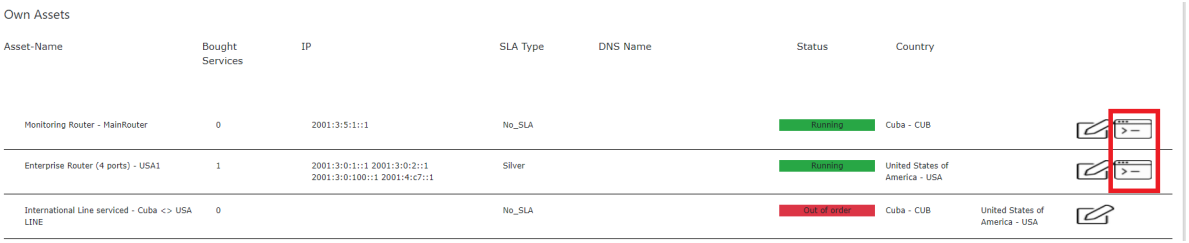


Abbildung 4.9.: Weiterleitung zur Virtualisierungsplattform

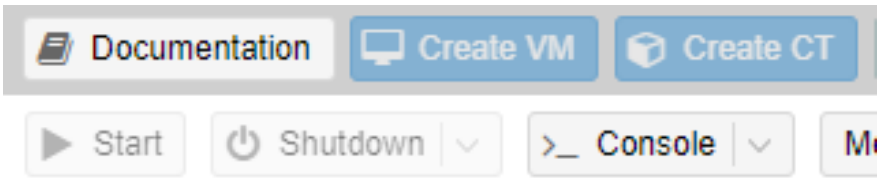
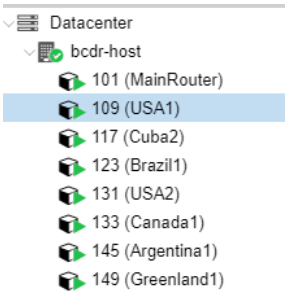


Abbildung 4.10.: Darstellung der eingeschränkten Berechtigungen auf der Virtualisierungsplattform

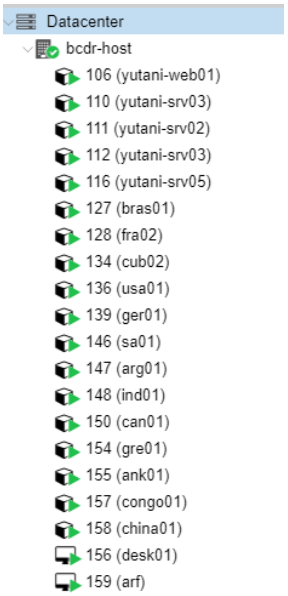
4.2.3. Simulation von Ausfällen

Der Ausfall von Komponenten wird jeden Tick zufällig, anhand einer angegebenen Ausfallrate, von der Spieloberfläche ermittelt. Um ausgefallene Komponenten für die Gruppen ersichtlich darzustellen, werden diese als „Out of Order“ markiert. In Abbildung 4.14 wird beispielsweise eine Netzwerkverbindung „International Line serviced“ als ausgefallen markiert.

Ein Ausfall von Servern, Routern und Desktop PCs wird simuliert, indem diese ausgeschaltet werden. Die Gruppen sind hier dafür verantwortlich, dass Konfigurationen nicht verloren gehen. Eine Simulation von Datenverlust oder beschädigten Datenträgern ist nicht implementiert. Alle erfolgreich gespeicherten Daten eines Servers oder Routers sind nach einem Ausfall wieder verfügbar. Durch die eingeschränkten Berechti-



(a) Beispiel der angezeigten Komponenten von Gruppe A



(b) Beispiel der angezeigten Komponenten von Gruppe B

Abbildung 4.11.: Gegenüberstellung der angezeigten Komponenten zweier Gruppen

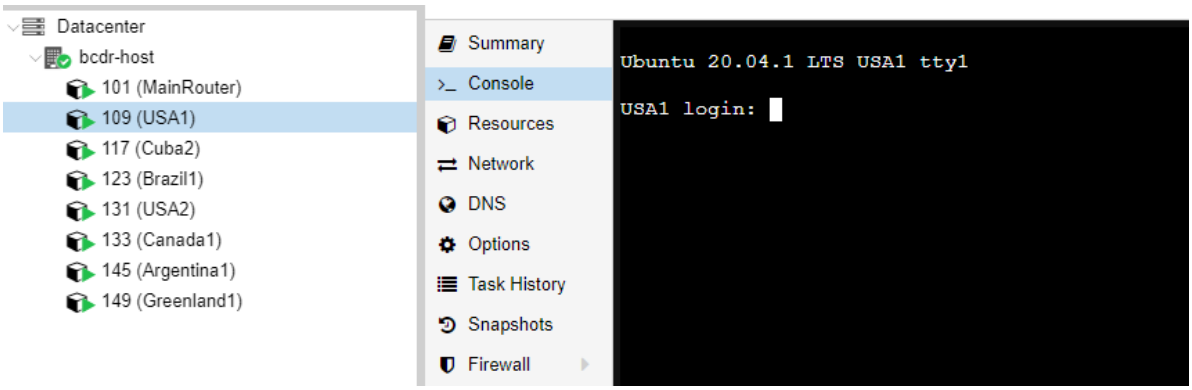
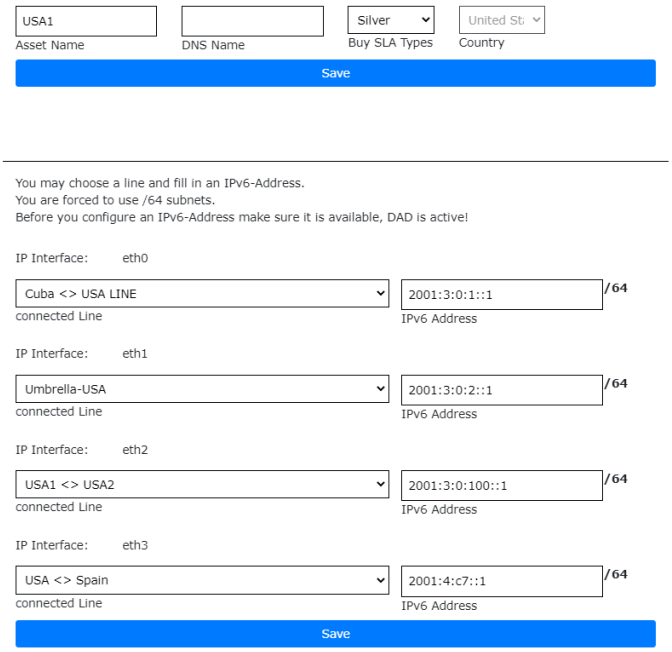


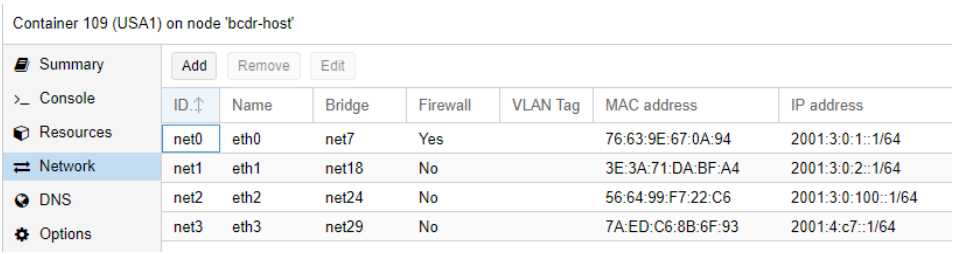
Abbildung 4.12.: Zugriff auf eine Komponente

gungen ist es einer Gruppe nicht möglich, ausgefallene Server, Router und Desktop PCs eigenständig wieder in Betrieb zu nehmen. Die Wiederinbetriebnahme der ausgefallenen Komponenten wird automatisch von der Spieloberfläche gesteuert. In Abbildung 4.15 sind in rot der Status („stopped“) eines ausgefallenen Servers und die deaktivierten Schaltflächen zum starten bzw. stoppen des Servers markiert.

Der Ausfall eines Netzwerkes konnte nicht durch das löschen des virtuellen Netzwerkes simuliert werden. Wird ein virtuelles Netzwerk gelöscht, so besteht es weiterhin bis alle Netzwerkinterfaces auf Server,Router und Desktop PCs entfernt wurden. Ein automatisches entfernen der Netzwerkinterfaces kann bestehende Konfigurationen der Gruppen auf ihren Komponenten verändern und wird daher vermieden.



(a) Netzwerkkonfiguration in der Spieloberfläche



(b) Netzwerkkonfiguration auf der Virtualisierungsplattform

Abbildung 4.13.: Darstellung der Netzwerkkonfiguration auf der Spieloberfläche und der Virtualisierungsplattform

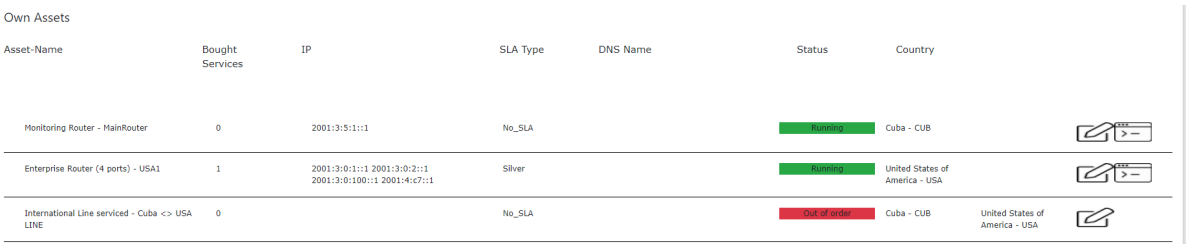


Abbildung 4.14.: Ausgefallene Komponente

Um einen Ausfall von Netzwerkverbindungen dennoch zu simulieren, wurde die durch die Virtualisierungsplattform bereitgestellte Firewall verwendet. Es wurde eine, in Abbildung 4.16 dargestellte, Gruppe an Firewall-Regeln erstellt. Diese Firewall-Regeln blockieren alle Arten von ein- und ausgehender Kommunikation, und werden automatisch an jedem Container (Router und Server) und jeder VM (Desktop PC) implementiert. Die Implementierten Firewall-Regeln werden in Abbildung 4.17 gezeigt.

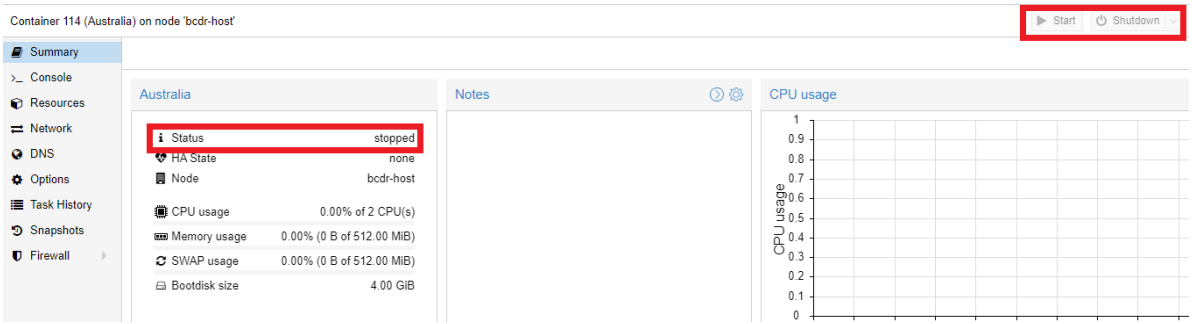


Abbildung 4.15.: Ansicht einer Gruppe auf einen ausgefallenen Server

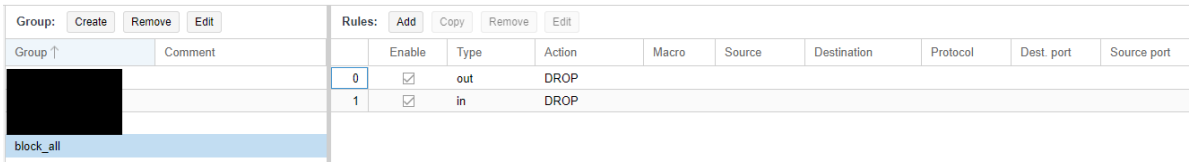


Abbildung 4.16.: Verwendete Firewall-Regeln

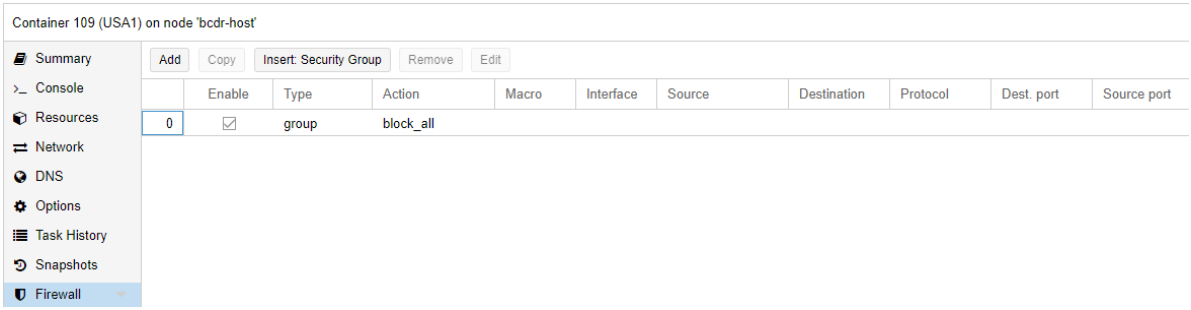
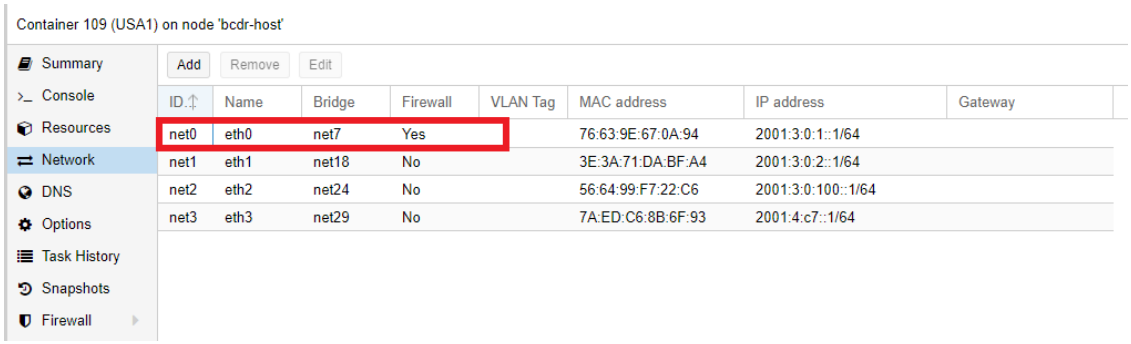


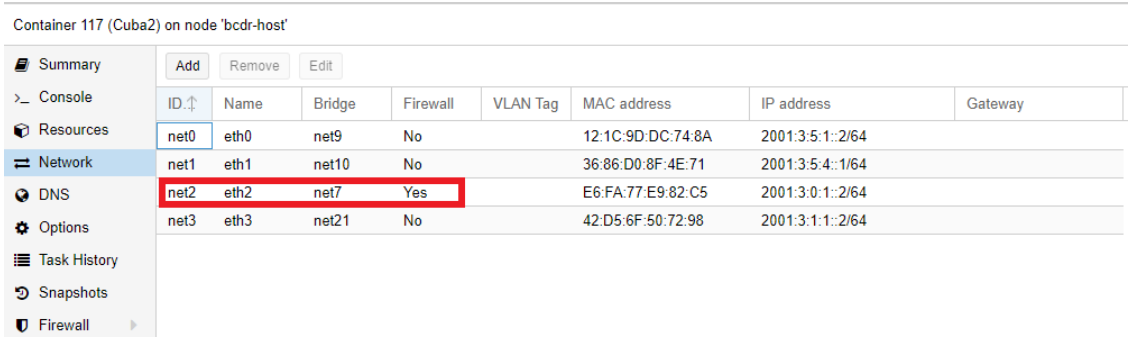
Abbildung 4.17.: Angewandte Firewall-Regeln auf einem Router

Die Firewall-Regeln können auf jedem Container und jeder VM auf jedem verwendetem Interface unabhängig voneinander aktiviert bzw. deaktiviert werden. Somit wird bei Ausfall einer Netzwerkverbindung von der Spieloberfläche jedes Interface ermittelt, welche mit dieser Netzwerkverbindung verbunden sind. Auf diesen Netzwerkinterfaces wird die Firewall aktiviert. Der dadurch erzielte Effekt ist, dass alle Kommunikation die durch diese Netzwerkverbindung gelangen sollte von den implementierten Firewall-Regeln verworfen werden. Damit kann ein Ausfall der Netzwerkverbindung simuliert werden. Um einen normalen Spielverlauf sicherzustellen ist standardmäßig die Firewall auf jedem Interface jedes Servers, Routers und Desktop PCs deaktiviert.

In Abbildung 4.18 ist dargestellt, dass die als „net7“ bezeichnete Netzwerkverbindung als ausgefallen simuliert wird, indem an beiden verbundenen Interfaces die Firewall aktiviert wurde. Bei der als „net7“ bezeichneten Netzwerkverbindung handelt es sich um die in Abbildung 4.14 als ausgefallen markierte Komponente. Dieser Zusammenhang kann durch Abbildung 4.19 und Abbildung 4.20 bestätigt werden.



(a) Ausgefallene Netzwerkverbindung auf Router „USA1“



(b) Ausgefallene Netzwerkverbindung auf Router „Cuba2“

Abbildung 4.18.: Ausgefallene Netzwerkverbindung auf zwei verschiedenen Routern

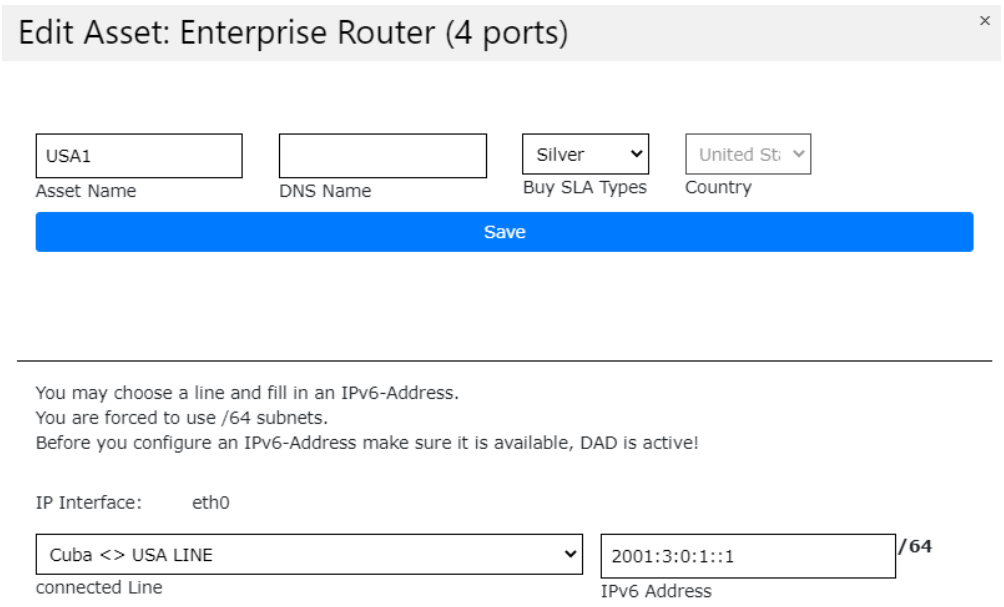


Abbildung 4.19.: Netzwerkkonfiguration von Router „USA1“

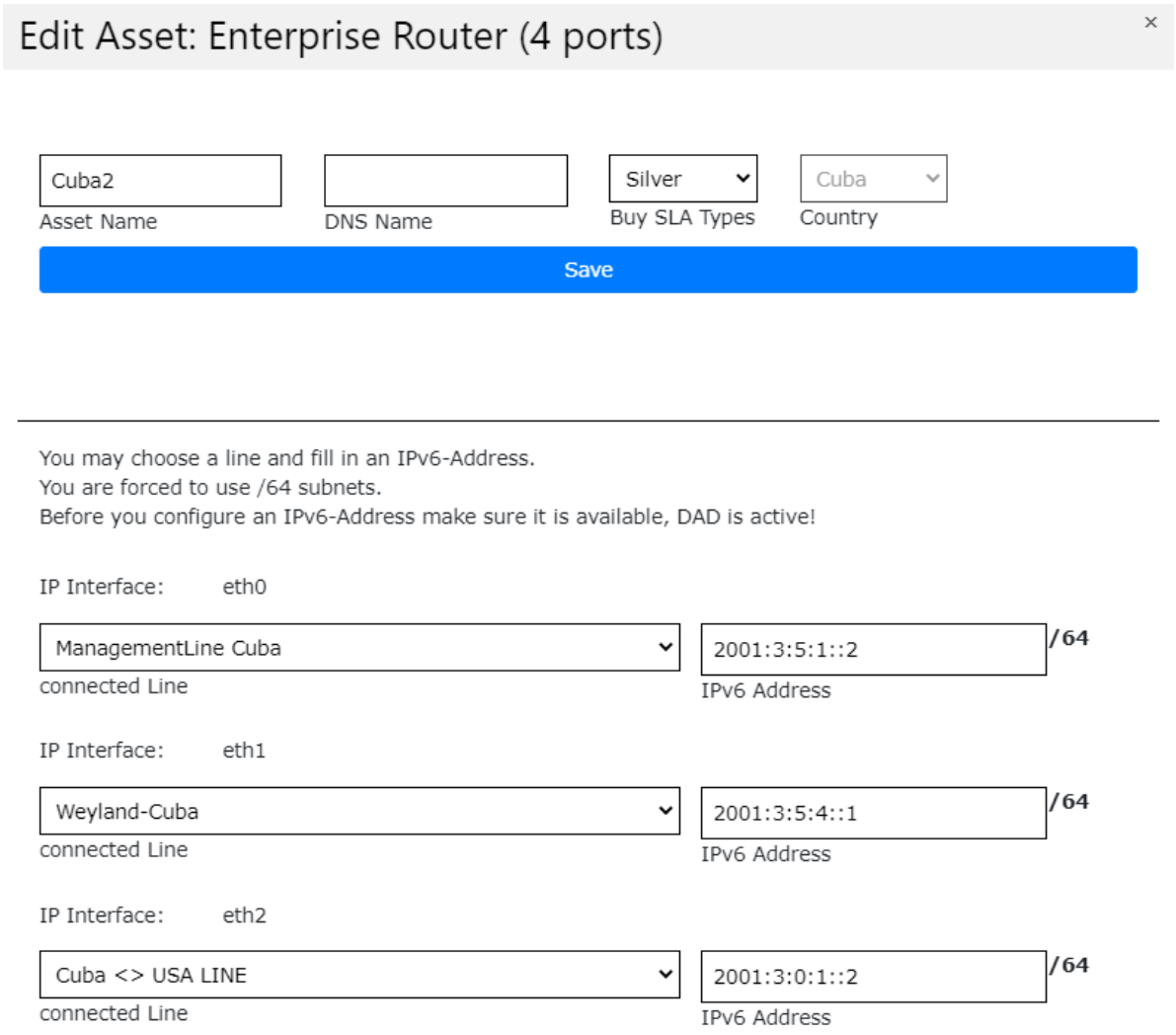


Abbildung 4.20.: Netzwerkkonfiguration von Router „Cuba2“

5. Conclusio

Im Zuge dieser Arbeit werden in Kapitel 3 verschiedene Technologien analysiert, welche zur Automatisierung und Virtualisierung von Netzwerkinfrastrukturen verwendet werden können. Vor allem Mininet [10] und Kathará [12] werden als potentielle Technologien, um automatisiert eine Netzwerkinfrastruktur zu erstellen, erkannt. Diese werden jedoch nicht eingesetzt, da sie nicht von mehreren Benutzer/inne/n verwendet werden können.

Zur Implementierung der Infrastruktur wird Proxmox VE [24] als Virtualisierungsplattform verwendet. Zur Verwaltung von virtuellen Netzwerken wird das noch in Entwicklung befindliche SDN Modul [45] von Proxmox VE verwendet. Server und Router werden mithilfe der LXC Containertechnologie [42] realisiert, diese wird direkt von der Virtualisierungsplattform unterstützt. Um den Gruppen zu ermöglichen alle geforderten Funktionalitäten zu realisieren, wurden Container Templates mit der dafür benötigten Software angefertigt. Das Softwarepaket FRR [66] wird verwendet, um eine erfolgreiche Kommunikation auf Netzwerkebene zu erreichen. Optionen zur Konfiguration von Komponenten wurden in die Spieloberfläche integriert, eine Schnittstelle zwischen Spieloberfläche und der Virtualisierungsplattform wurde in Form der von Proxmox VE zur Verfügung gestellten API [46] implementiert.

Die virtualisierte Infrastruktur wurde bereits zwei Mal in einem BCDR Planspiel verwendet. Die erste Durchführung des Planspiels mit der virtualisierten Infrastruktur fand im Januar 2021 statt. Teilnehmer/innen waren Studierende des Bachelorstudiengangs IT-Security der Fachhochschule St.Pölten. Die generelle Funktion der virtualisierten Infrastruktur war während des Spielverlaufes durchgängig gegeben. Einige Fehler wurden jedoch von Studierenden und Spielleiter/innen erkannt, und bereits teilweise während der Spieldurchführung behoben. Während der Durchführung traten Probleme am Monitoringsystem auf, welches durch den großen Zuwachs an zu überprüfenden Systemen zu langsam wurde. Dadurch wurden die Endresultate zu jedem Tick verfälscht. Es gelang die Performance des Monitoringsystems zu erhöhen um das Planspiel weiter durchzuführen, jedoch wird dieses Problem bei steigender Teilnehmerzahl erneut auftreten.

Das Ergebnis der ersten Durchführung des Planspiels war positiv, die virtualisierte Infrastruktur funktionierte wie gewollt. Im Zuge des Planspiels wurden ca. 25 Server, 25 Router und 60 Netzwerkverbindungen von den Gruppen erstellt und verwendet.

Die zweite Durchführung des Planspiels fand im Juni 2021 statt. Neben Studierenden des Bachelorstudiengangs IT-Security nahm auch eine Gruppe von Studierenden der Vidzeme University of Applied Sciences am Planspiel teil. Auch in diesem Durchgang des Planspiels funktionierte die virtualisierte Infrastruktur ohne größere Probleme. Es wurden erneut kleine Fehler gefunden, welche behoben wurden. Durch Feedback der Studierenden wurden Vorschläge zur Verbesserung von Spielmechaniken und zur Balance des im Spiel herrschenden Finanzmarktes eingebracht. Die Studierenden entdeckten auch einige Wege das Monitoringsystem zu überlisten, so wurden nicht umgesetzte Funktionalitäten als funktionierend markiert. Um das Ausnützen des Monitoringsystems zu verhindern, muss dieses aktualisiert und verbessert werden.

Im Zuge der zweiten Durchführung des Planspiels wurden 29 Server, 30 Router, 54 Netzwerkverbindungen und 2 Desktop PCs von den Gruppen erstellt und verwendet. Die Auslastung des verwendeten Servers, nachdem das Planspiel beendet wurde, jedoch mit noch bestehender Infrastruktur, zeigt Abbildung 5.1. Demnach werden weitere Durchführungen des Planspiels, auch mit größerer Zahl an Teilnehmer/innen keine Probleme durch zu großen Ressourcenverbrauch bereiten.

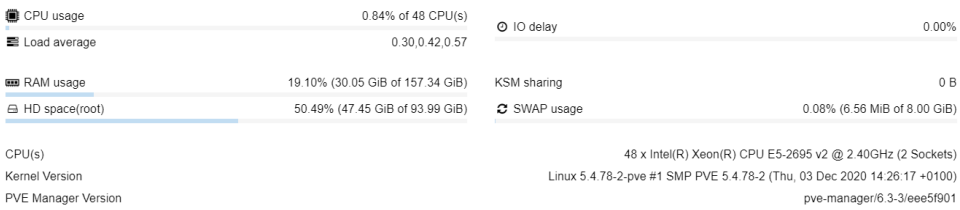


Abbildung 5.1.: Auslastung des Virtualisierungsservers

Die entwickelte virtualisierte Infrastruktur kann mit Erfolg eingesetzt werden. Einige in Abschnitt 5.1 beschriebene Probleme bestehen noch, welche durch vorübergehende Lösungen gelöst werden. Das Planspiel und das Monitoringsystem werden weiterentwickelt, daraus können einige in Abschnitt 5.2 beschriebene Arbeiten entstehen.

5.1. Einschränkungen

Da sich das SDN Modul [45] noch in der BETA Phase befindet, kann dieses noch Fehler beinhalten. Es ist möglich, dass sich die Implementation dieses Features auf der Virtualisierungsplattform Proxmox VE [24] noch in großem Ausmaß ändert. Eine ungetestete Verwendung dieses Moduls in Produktionsumgebungen kann ungewollte Ausfälle verursachen und deshalb sollte das Modul nur mit Vorsicht verwendet werden.

Um einen mögliche Beschränkung in der maximalen Anzahl an virtuellen Netzwerken zu überprüfen, wurde Testweise versucht möglichst viele virtuelle Netzwerke zu erstellen. Bei diesem Test konnten maximal 512 Netzwerke erstellt werden. Da keine offizielle Angabe zu einer maximalen Anzahl an Netzwerken in der Dokumentation des SDN Moduls [45] vorliegt, wurde dieses Testergebnis als Referenz herbeigezogen. Laut Proxmox Dokumentation liegt die maximale Anzahl an Netzwerkbrücken bei 4096, diese Angabe zieht die Verwendung des SDN Moduls nicht mit ein. [24] Eine Überschreitung dieser Grenze an möglichen Netzwerken bzw. Netzwerkbrücken ist im Zuge des Planspiels nicht wahrscheinlich.

Ein bekanntes und nicht behobenes Problem ist, dass die gesetzten Passwörter für Server und Router manchmal nicht richtig konfiguriert werden, und so den Gruppen der Zugriff auf den jeweiligen Server bzw. Router nicht möglich ist. Eine zwischenzeitliche Lösung dieses Problems ist es, den Spieleleiter/innen ein solches Problem zu melden. Die Spieleleiter/innen können, mithilfe der von der Virtualisierungsplattform bereitgestellten Programme [71], das Passwort eines Servers bzw. Routers neu setzen. Dieser Lösungsansatz wurde in den beiden Durchführungen des Planspiels erfolgreich angewandt.

Ein weiteres Problem ist, dass die Maximum Transmission Unit (MTU) bei einem Neustart eines Servers oder Routers verändert wird. Da verschiedene MTUs bei direkt verbundenen Geräten zu Kommunikationsproblemen führen [72], muss dieses Problem von den Studierenden selbständig behoben werden. Die MTU kann nach einem Neustart eines Servers/Routers auf den benötigten Wert zurück geändert werden. Die Gruppen werden zum Start des Planspiels auf dieses Problem hingewiesen um es schneller erkennen zu können.

Eine Einschränkung der implementierten Schnittstelle zwischen Spieloberfläche und Virtualisierungsplattform ist, dass nur ein Virtualisierungshost verwendet werden kann. Die Virtualisierungsplattform unterstützt zwar Cluster, jedoch können VMs und Container nur manuell oder mit entsprechender Logik zwischen ein-

zelnen Server des Clusters verschoben werden. [73] Diese erforderliche Logik wurde nicht in die Spieloberfläche implementiert.

5.2. Weiterführende Arbeiten

Die generelle Weiterentwicklung des Planspiels ist ein stetiger Prozess, nach jeder Durchführung werden die Studierenden um Feedback gebeten. Einfach umzusetzende Änderungen werden so schnell als möglich in die Dokumente und den Programmcode des Planspiels eingearbeitet.

Sobald das SDN Modul von Proxmox VE als stabil markiert wird, soll die Virtualisierungsplattform auf die stabile Version upgedatet werden. Sollten dabei Änderungen in verwendeten Funktionen der Proxmox VE API vorliegen, muss die Spieloberfläche dahingehen angepasst werden. Die Verwendung eines experimentellen Features trägt immer Risiken mit sich, diese können mit einem Update auf als stabil markierte Pakete minimiert werden. Wenn als stabil markierte Pakete verwendet werden, könnten auch in Abschnitt 5.1 aufgezählte bekannte Probleme gelöst werden.

Da in den letzten beiden Durchführungen des Planspiels beide male Probleme mit dem Monitoringsystem erkannt wurden, und dieses schon längere Zeit ohne Weiterentwicklung in Betrieb ist. Sollte das Monitoringsystem zumindest komplett überarbeitet und verbessert werden. Eine Neuimplementierung des Systems könnte vorhandene Probleme beseitigen, jedoch können damit auch neue Probleme entstehen.

Aktuell werden geplante „Desaster“ wie in [1] beschrieben nur manuell durchgeführt. Die Durchführung dieser kann automatisiert werden, so können auch komplexere Desaster ermöglicht werden. Ein Menü für Spielleiter/innen zur Erstellung und Durchführung von Desastern kann in die Spieloberfläche implementiert werden.

Um die Studierenden weiter zu fordern, können einige Router und Server Templates mit verschiedenen Sicherheitslücken erstellt werden. Sobald ein neuer Router oder Server von einer Gruppe erstellt wird, kann zufällig eines dieser Templates verwendet werden. Ziel der Studierenden soll es sein, die Sicherheitslücke zu finden und zu schließen. Die Ausnutzbarkeit einer Sicherheitslücke könnte auch von dem bereits bestehendem Monitoringsystem überprüft werden, wenn dieses dazu weiterentwickelt wird. Geschlossene Sicherheitslücken werden mit Boni an die Gruppe belohnt.

Soll das Planspiel kompetitiver gestaltet werden, können zusätzliche Aufgaben an die Gruppen verteilt werden. Als Beispiel kann ein Bonus für die erste Gruppe ausgezahlt werden, welche ein gewisses Land erschließt. Diese Aufgaben könnten in der Spieloberfläche angezeigt, und von den Spielleiter/innen verwaltet werden.

Einige neue Features können der Spieloberfläche hinzugefügt werden:
Verträge die zwischen Gruppen geschlossen werden, sollten mittels OpenPGP [74] signiert werden. Damit sollen die Studierenden mit OpenPGP vertraut gemacht werden. Die Signaturen sollen automatisch auf der Spieloberfläche verifiziert werden. Aktuell können Verträge zwischen Gruppen nur mittels Schaltflächen akzeptiert oder verworfen werden.

Um den Studierenden einige erreichte Ziele aufzuzeigen, soll eine Ansicht mit verschiedenen Statistiken zum Spielverlauf implementiert werden. Diese kann zum Ende des Planspieles von den Spielleiter/innen verwendet werden um den Gruppen besseres Feedback zu geben. Einige Beispiele für mögliche Statistiken:

- Anzahl von verwendeten Server, Router, Netzwerkverbindungen, Desktop PCs und deren Unterkategorien pro Gruppe.
- Anzahl der verwendeten Service Level Agreement (SLA)s pro Gruppe.
- Abgeschlossene Verträge unter den Gruppen
- Erste Gruppen pro Kontinent
- Zeit zur bis zur erfolgreichen Implementierung einer geforderten Funktionalität

Die GUI der Spieloberfläche kann überarbeitet werden, um die Benutzung zu vereinfachen und das Planspiel moderner wirken zu lassen.
Die in Abschnitt 4.1 beschriebene Funktionalität zur Erstellung von VMs kann in Zukunft für neue Spielmechaniken genutzt werden. Um VMs zu erstellen können die Gruppen Komponenten mit dem Titel „Desktop PC“ kaufen. Diese Komponente kann verwendet werden, da sie zur Zeit keinen weiteren Nutzen hat. Als Beispiel kann die eigentliche Funktion, um Router zu konfigurieren, mit dieser Komponente realisiert werden.
Die in Abbildung 4.9 gezeigte Einbindung des Zugriffes auf Server, Router und Desktop PCs kann überarbeitet werden. Mithilfe der API von Proxmox VE ist es möglich, direkt Konsolenfenster in die Spieloberfläche einzubinden. Dadurch wäre eine Weiterleitung, und ein erneutes Einloggen, auf die Virtualisierungsplattform nicht nötig. So kann der Spielfluss erleichtert, und die Spielerfahrung verbessert werden.

A. General Guide

Business Continuity and Disaster Recovery

The disconnected days

André Meindorfer (original Author)
is131014@fhstp.ac.at

Benjamin Petermaier (original Author)
is131018@fhstp.ac.at

Georg Graf (changes 2017-06)
141016@fhstp.ac.at

Harald Rezmann (changes 2017-06)
141031@fhstp.ac.at

Martin Valicek (changes 2017-06)
141039@fhstp.ac.at

Bernadette Jilch (changes 2020-06)
is191816@fhstp.ac.at

Stefan Machherndl (changes 2021-05)
is191812@fhstp.ac.at

May 27, 2021

Dedicated to our Internet and in memory of legacy IP[a].

[a] formerly known as IPv4

Abstract

This document describes all prevailing rules for the BCDR[1] crisis simulation. Read this rule set carefully to prevent you and your team from major disadvantages.

To achieve ideal experiences this exercise is organized as a simulation game wrapped around a comprehensive lab exercise. All participants, organized in groups with different tasks, work together towards a common goal, but still compete against each others. The setting is a model of the real world Internet with various economical aspects.

The whole set of participants' goal is to rebuild the Internet as fast as possible, groups strive towards power, glory and honor, and each individuals goal is to earn as much ₤[2] as possible. Each individuals final monetary resources define the grade.

We did our best to balance the rules for the game, still some rules might be erroneous or impractical, based on the course of the exercise. In this situations, a board consisting of lecturers and participants is formed, deciding about changes.

[1] Business Continuity and Disaster Recovery
[2] Labcoin

Contents

1

Grading

The lab exercise is organized as economical simulation to encourage competition. More advantages over other teams lead to better grading. At the end of the game all assets of the team are exchanged back into their acquisition value, then each teams financial resources are spread evenly across all team members. Finding grades is easy - the participant with the highest amount of £ gets grade 1, the participant with the least amount of £ gets grade 3. Participants in between are scaled according to their wealth. Worse grades (4-5) are only assigned for plain disinterest, absence or similar.

2

Story

As for every decent role play game this one also has a background story of some importance. In the following paragraphs this story is shortly recited.

2.1 Background

When a massive solar flare hit the atmosphere it had an effect similar to an EMP to all electronic devices. Most of the devices could be brought back up in a short period of time, but the complexity and sluggishness of the Internet made it so that although a huge effort went into trying to bring it back up, it never made it back. This incident has an impact on the majority of the earths population. People are disoriented, angry and afraid. But besides the obvious loss of exceptionally cheap entertainment, mankind faces other, more serious problems. For instance many live-critical systems are heavily dependent on the Internet.

Leading representatives of the industry agree on the topic of rebuilding the Internet as fast as possible and see this as a chance to abandon deprecated technologies such as legacy IP[1] and unencrypted connections. The bravest and best companies in their respective field of expertise take on this nigh impossible task.

2.2 The Task

The team in charge of rebuilding the Internet is composed of four corporations providing security-specific services and three Internet service providers for basic communication. The four companies agree on splitting the task of domain services and certificate based services. Two companies operate as official certificate authorities, the other two as domain service providers.

The Internet service providers, located at three different continents on

[1] IPv4

the northern hemisphere, try to split the traffic between them and reach a failure resistant, converged network as fast as anyhow possible.

3

Rules

The objective of the game is to score as much £ as possible until the end of the game. This defines the final grade for this subject.

The game is divided in hourly ticks, each consisting of one hour. At the end of each tick the groups financial resources are recalculated, ROI[1] is calculated and failures are identified. The amount of ticks per day is announced by the game-masters at the beginning of the game.

There is an extra scoring with the purpose of comparing different classes over different years. This final score is not part of the grade, it is just an addition that is supposed to make the game a little more interesting and competitive.

3.1 Commerce

As mentioned before, the game is an economical simulation, therefor money and commerce play a big role in both the game itself and the grading. The following paragraphs describe the general approach of commerce in the game.

3.1.1 Initial funds

Each team starts with an initial fund of 17,000£.Money can be freely moved between teams.

3.1.2 Return On Investment

Every asset purchased by a team will bring said team a ROI per tick. The rate for ROI is 4% of the initial, one-time payment. Recurring payments are not part of the ROI. If an asset is damaged and out-of-order, this asset will not gain ROI.

[1] Return on Investment

3.1.3 Periodic payout

A main source of income for a team is the periodic payout. This payout consists of the ROI and a fixed value per point of presence (see section 3.3).

The periodic payout is also based on a phase system. Each team has certain milestones to reach, and with every higher stage, the periodic payout increases. The teams start in Phase 0 and will get 0% of the periodic payout, when they reach Phase 1 they will get 100% and in the final Phase 2 each teams gets 200% of the periodic payout. This means the periodic payout does not kick in until you have reached Phase 1!
The formula for the periodic payout is:

ROI = initial asset value * 0.04 * phase * uptime[%]

POP Bonus = (sum(PoP)*250£) * phase

Please note that you only gain the ROI only once per asset per tick, if you have more services running on one asset the service with the biggest uptime is considered in the calculation of the ROI.

3.1.4 Contracts

The second big source of income are contracts with other teams. As this is an economic simulation, every team has to provide and sell services and goods. The deals between groups are only valid if there is a valid contract provided to the game-masters. The content of the contracts can be freely chosen, just keep in mind that they must be fulfilled in time.

Hand in the finalized contracts to the game-masters.

3.2 Dead Body Count

This extra goal is to encourage the class as a whole to do their best and work together as good as possible. Each team that is not in Phase 2 at the end of a tick(see individual team descriptions in section 4) causes people to loose their lives.

The idea is that out of 8 Billion people on the planet earth every single one is dead if no team manages to get out of Phase 0 the entire game. To the contrary every one survives if every team is in Phase 2 all game long. Your classes final score will be published proudly somewhere in the NetLABs for everyone to admire.

To avoid any casualties all teams have to deliver outstanding performance. Your time has come to save lives.

3.3 Point of Presence

For maximising redundancy, reliability and also profit a team may want to open multiple business locations. These locations count as Point of Presence

and add to the periodic payment described in section 3.1.3.
There can only be one PoP per Country per Team in one Country. This means, having two Server or Router in one Country does not generate two or more PoPs.

To be eligible as Point of Presence, the business locations need to reach the following requirements:

For a company a PoP needs to be connected to the Internet and have at least one running service.

For an ISP a PoP needs to be connected to the own network and have at least one router present.

3.3.1 POP definition for ISPs

At least one routing service assigned to this POP must be active. The routing service must exist for your own (not your customers) use and the monitoring must at least define it as 'basic-up'.

3.3.2 POP definition for Companies

At least one of the services the company has to provide must exist and monitored as 'basic-up'. The service(s) must be registered to the particular team and country.

3.4 Reaching a new continent

The first companies and ISPs to reach an empty[2] continent and establish a PoP are eligible to a bonus. This bonus is 15,000£ for ISPs and 10,000£ for companies. If two or more companies reach the same continent at the same tick, the full bonus will be given to all parties. If two or more ISPs reach the same continent at the same tick, the bonus is equally distributed between the ISPs in question.

This bonus is not automatically distributed. You will have to approach the game-masters to claim it.

3.5 World map

To avoid any confusions and discussions, the set of countries and continents and respective continental borders can be found when buying new assets or in the world map which is available on the groups management site (Dashboard). Continents or countries outside of this list are non existent - we checked it, they did not survive. All dead.

[2] empty: no other team of your kind (ISP, Company) has a POP on this continent

3.6 Buying assets

Infrastructural assets can be bought in differing quality. This quality is mirrored in the failure probability of the asset.

To host a specific service (mail-server, dns-server, web-server, ...) the according hardware is needed. This hardware has to be purchased from the game-masters at rates according to the table at section 3.6.2. The connections and lines between assets also have to be purchased from the game-masters. Lines already have built in switches (technology moves fast), therefore multiple assets can be connected to one line on each side of the line.

It is possible to buy Service Level Agreements for assets to decrease the chance of failure of this particular asset. Only one SLA per asset may be purchased.

Every Team can only buy Hardware fitting to its team type.

3.6.1 Failures

The failure of an asset is determined by luck (dice roll) and its failure probability. The failure probability describes the chance of the service to fail in the following tick. If an asset fails, it may not be used in the following tick whatsoever. Teams must not use this asset in any way (e.g. copying files, making backups, operate devices connected to failed line, ...).

If an SLA is present for the failed asset, as a safety net, the failure probability of this SLA is determined. The SLA can also fail, causing the asset to fail anyway.

3.6.2 Products

The following table contains all products available to the teams. Some of the products do have recurring costs besides the initial purchase cost. When buying these products a certain value of the investment is coming back to the buying party as Return On Investment. For further information on ROI see section 3.1.

product	failure %	one-time cost	running costs
Desktop PC	9%	1,000£	-
Small Server (1 Application)	5%	2,000£	-
Medium Server (3 Applications)	4%	8,000£	-
Large Server (6 Applications)	2%	16,000£	-
Hypervisor (9 Applications)	1%	24,000£	-
SLA Silver	25%	-	250£
SLA Gold	5%	-	1000£
National Line	7%	1,000£	100£
	4%	1,000£	200£
International Line	9%	2,000£	200£
	5%	2,000£	350£
Intercontinental Line	2%	5,000£	500£
Router (2 Ports)	5%	1,000£	-
Enterprise Router (4 Ports)	2%	3,000£	-

Table 3.1: List of assets

3.6.3 Lines

For connecting sites, certain lines can be purchased. There are different types for connecting devices nationally, internationally and intercontinentially. To connect a server of an company to an ISP router both properties MUST be in the same country and a national line is required. Routers in adjacent countries can only be connected using international lines. Connections that connect countries of different continents must be realized using intercontinental lines.

These lines are handled the same as other assets, meaning the lines can fail, can be insured by a SLA and the purchasing party get ROI for bought lines.

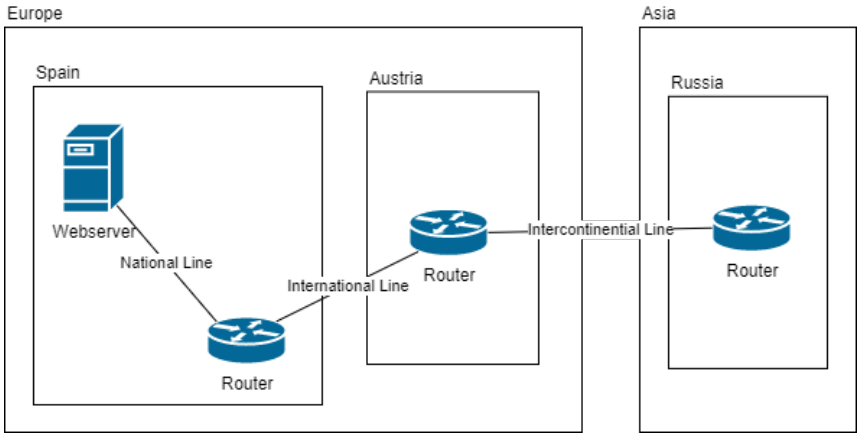


Figure 3.1: Example for a line configuration

3.7 Prohibitions

Excluded from the hacking paragraph is the game infrastructure as well as the game supporting web application. A malicious activity on said components leads to an instantaneous exclusion of the course and a negative grading.

When you happen to find a vulnerability in the infrastructure please report to the game-masters.

4

Teams

Working in teams is fun and this game is designed to be a team game. Every student is part of a team of more or less three people. In the following paragraphs the different types of teams and each team in particular are described.

4.1 Corporations

The corporations provide the services TLS and DNS which are of course integral parts of the new (and also old) Internet. Every team is dependent on this two services so the corporations got to be fast in implementing them. There are two teams responsible for TLS and two for DNS. This allows no monopoly position of any kind.

Phases

The phases according to 3.1.3 are as following

- Phase 1: A webshop to display your goods reachable via IP
- Phase 2: Connect your webshop and a mailserver to the internet. The webshop must be secured via TLS, and reachable via DNS and secured via DNSSEC.

The webshop does not have to have an actual billing system. It serves the purpose of displaying your goods, no further.

4.1.1 Aperture Science, Inc.

Aperture Science resides in California, USA. It's core business, besides weird experiments with artificial intelligence and robotics, is the implementation of a domain name system and selling domains to other parties. Their main competitor is the Black Mesa Research Facility located in Saudi Arabia.

Task

Your task is to setup a globally functioning domain name system in cooperation with the competitor.
Note: All your customers need to alter their root hints, you may want to provide them an easy way to do so.

4.1.2 Weyland-Yutani Corporation

The new rising star on the startup scene, Weyland-Yutani (short Wey-Yu), based in South Africa, participate in rebuilding the Internet as provider of certificate based services, in particular building a global certificate authority and selling SSL certificates to customers. Because of the strong competition, coming from the Umbrella Corp. in Europe, it is rumoured, that the company founders of Wey-Yu are also trying to get hold in the field of space exploration.

Task

Your task is to set up certificate services to be available for everyone. Together with your competitor, create an environment with two root CAs for all needed certificate services.
Note: You may want to provide an easy way for your customers to import your certificate in their certificate stores.

4.1.3 Black Mesa Corporation

The Black Mesa Corporation primes on the field of IT-Services and interdimensional research. It's main compound is the Black Mesa Research Facility in the country of Saudi Arabia. Together with Aperture Science in the US, Black Mesas vision is to set up a functioning DNS in the very near future.

Task

Your task is to setup a globally functioning domain name system in cooperation with the competitor.
Note: All your customers need to alter their root hints, you may want to provide them an easy way to do so.

4.1.4 Umbrella Corporation

Besides Weyland-Yutani, Umbrella Corp. is the biggest new provider of certificate based services. Originally founded in the US, they were fast to change their location to Sweden. It is said that this was due to failed business

affairs in their former field of study, which included biology in a city named after some aggressive mammal.

Task

Your task is to set up certificate services to be available for everyone. Together with your competitor, create an environment with two root CAs for all needed certificate services.
Note: You may want to provide an easy way for your customers to import your certificate in their certificate stores.

4.2 Internet Service Provider

The backbone of the new-to-be Internet are, of course, the ISPs. Three companies have come together to agree on terms for the newly created network. These three, all residing on different continents, are as following

- Level 3 Communications - Cuba, NORTH AMERICA
- TeliaSonera International Carrier - France, EUROPE
- Akamai Technologies - Russia, ASIA

These companies have to provide the most integral part of the Internet: a converged network. Every other team is heavily dependent on the work of the ISPs.

Phases

The phases according to 3.1.3 are as following

- Phase 1: Provided Routing Service for Customer
- Phase 2: additional Webshop and Mailserver hosted by any Corporation, the webshop must be secured via TLS, and reachable via DNS and secured via DNSSEC.

A high responsibility weighs on your shoulders. The Internet will not work if you're not in Phase 1 fast. Try to work as hard as you can, to reach a fully converged net.

Hints

On Proxmox (the virtualization platform of the whole infrastructure) you can login using the same credentials you got for the game site, here you can see all your Machines and also get a shell for them. (As Realm use "Proxmox VE authentication server")

The IPv6-Address of the Monitoring Server is 2001:1::babe:1, you can use it to check if at least the connection to it is working.

Known Problems

Sometimes the configured root password is not working for a newly spawned machine, to fix that sell the new spawned machine and buy one again, you can also report it to the game-masters and they will set the password to a default one.

When machines restart (or get up again after failing), they sometimes change the MTU size on some interfaces. If routers can't find each other after one restarted check the MTU and make sure it is the same on the connected interfaces. (as default the machines use a MTU of 1450)

For Companies:
On the server image the following apt packages are preinstalled (if you still need some more report that to the game-masters, if they feel gracious they will update the images, so that any new bought servers will have the needed packages)

- iproute2
- openssh-server
- supervisor
- apache2
- postfix
- rsyslog
- bind9
- bind9utils
- bind9-doc
- traceroute
- iputils-ping
- vim
- nano

- php
- wget
- curl
- tcpdump
- dnsutils
- netcat-openbsd
- telnet

For ISPs

The following will be helpful:
FRR Routing Documentation

To enter a Cisco like router shell enter the following in the command line of your router:
vtysh

The subnets 2001:1::/64 and 2001:2::/48 are not available for you, all other subnets you can use as you wish. (maybe talk to other ISPs first)

The MainRouter is connected to a router operated by the game-masters, you need to use it to get a connection to the monitoring server. (Use bgp to share your routes with the game-masters router)
The MainRouter will be up all the time and cannot fail, the lines from it to the game-masters router will also not fail.

Router RAs are blocked by the firewall, so make sure to tell your customers what they should use as default gateway. (and DNS-Server if they don't know)

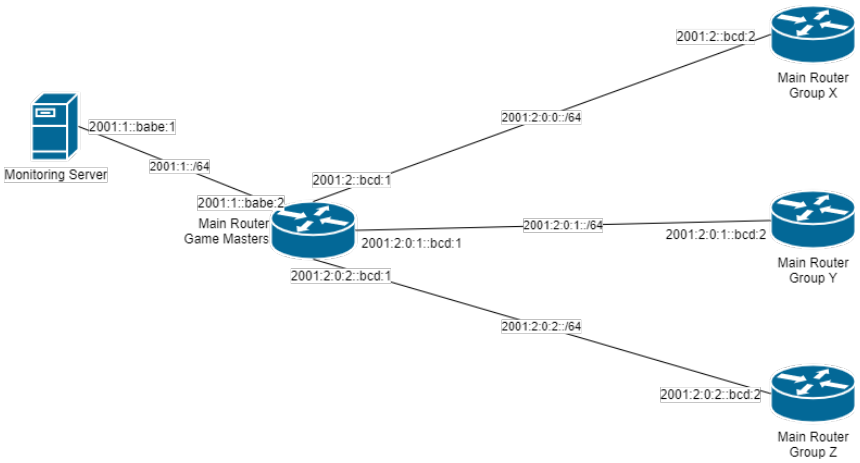


Figure 4.1: Connections between game-master router and the routers of the ISPs

Abbildungsverzeichnis

Tabellenverzeichnis

Glossar

API	Application Programming Interface
BCDR	Business Continuity and Desaster Recovery
CML	Cisco Modeling Labs
DNS	Domain Name System
GUI	Graphical User Interface
IPv4	Internet Protocol Version 4
IPv6	Internet Protocol Version 6
MTU	Maximum Transmission Unit
PKI	Public-Key-Infrastruktur
SDK	Software Development Kit
SDN	Software-defined Networking
SLA	Service Level Agreement
VIRL	Virtual Internet Routing Lab
VM	Virtuelle Maschine

Literatur

[1] Christoph Lang-Muhr Daniel Haslinger, "Business Continuity & Desaster Recovery als Planspiel umgesetzt", in *Tagungsband zum 5. Tag der Lehre an der FH St. Pölten am 20.10. 2016*, 2016, S. 117–125.

[2] VMware. "VMware Workstation", Adresse: `https://www.vmware.com/products/workstation-pro.html` (besucht am 25. 07. 2021).

[3] "GNS3", Adresse: `https://www.gns3.com/` (besucht am 25. 07. 2021).

[4] Bruce Perens u. a., "The open source definition", *Open sources: voices from the open source revolution*, Jg. 1, S. 171–188, 1999.

[5] Cisco. "Cisco CSR 1000V", Adresse: `https://www.cisco.com/c/en/us/products/routers/cloud-services-router-1000v-series/index.html` (besucht am 20. 08. 2021).

[6] ——, "Cisco IOS XE", Adresse: `https://www.cisco.com/c/en/us/products/ios-nx-os-software/ios-xe/index.html` (besucht am 20. 08. 2021).

[7] Christoph Seifert, Sven Reißmann, Sebastian Rieger und Christian Pape, "Evaluation von VIRL, GNS3 und Mininet als Virtual Network Testbeds in der Hochschullehre", in *11. DFN-Forum Kommunikationstechnologien*, Paul Müller, Bernhard Neumair, Helmut Reiser und Gabi Dreo Rodosek, Hrsg., Bonn: Gesellschaft für Informatik e.V., 2018, S. 103–112.

[8] Cisco. "Cisco Packet Tracer", Adresse: `https://www.netacad.com/courses/packet-tracer` (besucht am 05. 09. 2021).

[9] Nikhil Handigol, Brandon Heller, Vimalkumar Jeyakumar, Bob Lantz und Nick McKeown, "Reproducible network experiments using container-based emulation", in *Proceedings of the 8th international conference on Emerging networking experiments and technologies*, 2012, S. 253–264.

[10] Mininet Project Contributors. "Mininet", Adresse: `http://mininet.org/` (besucht am 05. 09. 2021).

[11] Cisco. “Cisco Modeling Labs”, Adresse: https://www.cisco.com/c/en/us/products/cloud-systems-management/modeling-labs/index.html (besucht am 05. 09. 2021).

[12] Research group from Roma Tre University. “Kathará”, Adresse: https://www.kathara.org/ (besucht am 05. 09. 2021).

[13] Gaetano Bonofiglio, Veronica Iovinella, Gabriele Lospoto und Giuseppe Di Battista, “Kathará: A container-based framework for implementing network function virtualization and software defined networks”, in *NOMS 2018 - 2018 IEEE/IFIP Network Operations and Management Symposium*, 2018, S. 1–9. DOI: 10.1109/NOMS.2018.8406267.

[14] Mariano Scazzariello, Lorenzo Ariemma und Tommaso Caiazzi, “Kathará: A Lightweight Network Emulation System”, in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, S. 1–2. DOI: 10.1109/NOMS47738.2020.9110351.

[15] Docker, Inc. “Docker”, Adresse: https://www.docker.com/ (besucht am 11. 09. 2021).

[16] The Kubernetes Authors. “Kubernetes”, Adresse: https://kubernetes.io/ (besucht am 11. 09. 2021).

[17] Mariano Scazzariello, Lorenzo Ariemma, Giuseppe Di Battista und Maurizio Patrignani, “Megalos: A Scalable Architecture for the Virtualization of Network Scenarios”, in *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, 2020, S. 1–7. DOI: 10.1109/NOMS47738.2020.9110288.

[18] Mallik Mahalingam, Dinesh Dutt, Kenneth Duda, Puneet Agarwal, Larry Kreeger, T. Sridhar, Mike Bursell und Chris Wright, *Virtual eXtensible Local Area Network (VXLAN): A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks*, RFC 7348, Aug. 2014. DOI: 10.17487/RFC7348. Adresse: https://rfc-editor.org/rfc/rfc7348.txt.

[19] Docker, Inc. “Docker Networking overview”, Adresse: https://docs.docker.com/network/ (besucht am 11. 09. 2021).

[20] Oracle. “VirtualBox”, Adresse: https://www.virtualbox.org/ (besucht am 11. 09. 2021).

[21] VMware. “VMware vSphere”, Adresse: https://www.vmware.com/products/vsphere.html (besucht am 12. 09. 2021).

[22] KVM contributors. “KVM”, Adresse: https://www.linux-kvm.org/page/Main_Page (besucht am 12. 09. 2021).

[23] Microsoft. “Hyper-V”, Adresse: https://docs.microsoft.com/en-us/windows-server/virtualization/hyper-v/hyper-v-technology-overview (besucht am 12. 09. 2021).

[24] Proxmox Server Solutions GmbH. “Proxmox VE”, Adresse: https://www.proxmox.com/en/proxmox-ve (besucht am 20. 08. 2021).

[25] Microsoft. “Microsoft Info for the Gartner Magic Quadrant for Server Virtualization”, Adresse: https://info.microsoft.com/Gartner-MQ-Server-Virtulization.html?ls=Blogl (besucht am 12. 09. 2021).

[26] Thomas J Bittman, George J Weiss, Mark A Margevicius und Philip Dawson, “Magic quadrant for x86 server virtualization infrastructure”, *Gartner, June*, 2013.

[27] VMware. “VMware ESXI”, Adresse: https://www.vmware.com/products/esxi-and-esx.html (besucht am 11. 09. 2021).

[28] ——, “VMware vCenter Server”, Adresse: https://www.vmware.com/products/vcenter-server.html (besucht am 11. 09. 2021).

[29] ——, “VMware NSX Data Center”, Adresse: https://www.vmware.com/products/nsx.html (besucht am 12. 09. 2021).

[30] ——, “VMware vSphere with Tanzu”, Adresse: https://www.vmware.com/products/vsphere/vsphere-with-tanzu.html (besucht am 12. 09. 2021).

[31] ——, “VMware vSphere Automation SDK Documentation”, Adresse: https://www.vmware.com/support/pubs/sdk_pubs.html (besucht am 12. 09. 2021).

[32] QEMU developers. “QEMU”, Adresse: https://www.qemu.org/ (besucht am 12. 09. 2021).

[33] KVM contributors. “KVM Management Tool List”, Adresse: https://www.linux-kvm.org/page/Management_Tools (besucht am 12. 09. 2021).

[34] Microsoft. “Windows Server”, Adresse: https://www.microsoft.com/en-us/windows-server (besucht am 12. 09. 2021).

[35] ——, “Windows 10”, Adresse: https://www.microsoft.com/en-us/windows/windows-10-specifications (besucht am 12. 09. 2021).

[36] ——, “Virtual Switch for Hyper-V”, Adresse: https://docs.microsoft.com/en-us/windows-server/virtualization/hyper-v/get-started/create-a-virtual-switch-for-hyper-v-virtual-machines (besucht am 12. 09. 2021).

[37] ——, “Hyper-V Network Virtualization”, Adresse: https://docs.microsoft.com/en-us/windows-server/networking/sdn/technologies/hyper-v-network-virtualization/hyperv-network-virtualization-overview-windows-server (besucht am 12. 09. 2021).

[38] ——, “Windows and containers”, Adresse: https://docs.microsoft.com/en-us/virtualization/windowscontainers/about/ (besucht am 12. 09. 2021).

[39] ——, “Hyper-V APIs”, Adresse: https://docs.microsoft.com/en-us/virtualization/api/ (besucht am 12. 09. 2021).

[40] Software in the Public Interest, Inc. “Debian”, Adresse: https://www.debian.org/index.en.html (besucht am 12. 09. 2021).

[41] Proxmox Server Solutions GmbH. “Support for Proxmox VE”, Adresse: https://www.proxmox.com/en/proxmox-ve/support (besucht am 12. 09. 2021).

[42] LinuxContainers. “LXC”, Adresse: https://linuxcontainers.org/ (besucht am 20. 08. 2021).

[43] The kernel development community. “Virtual eXtensible Local Area Networking documentation”, Adresse: https://www.kernel.org/doc/html/latest/networking/vxlan.html (besucht am 12. 09. 2021).

[44] ——, “Virtual Routing and Forwarding (VRF)”, Adresse: https://www.kernel.org/doc/html/latest/networking/vrf.html (besucht am 12. 09. 2021).

[45] Proxmox Server Solutions GmbH. “Proxmox SDN”, Adresse: https://pve.proxmox.com/pve-docs/chapter-pvesdn.html (besucht am 20. 08. 2021).

[46] ——, “Proxmox API Documentation”, Adresse: https://pve.proxmox.com/pve-docs/api-viewer/ (besucht am 20. 08. 2021).

[47] Stephen Soltesz, Herbert Pötzl, Marc E Fiuczynski, Andy Bavier und Larry Peterson, “Container-based operating system virtualization: a scalable, high-performance alternative to hypervisors”, in *Proceedings of the 2Nd ACM SIGOPS/EuroSys european conference on computer systems 2007*, 2007, S. 275–287.

[48] Roberto Morabito, Jimmy Kjällman und Miika Komu, “Hypervisors vs. Lightweight Virtualization: A Performance Comparison”, in *2015 IEEE International Conference on Cloud Engineering*, 2015, S. 386–393. DOI: 10.1109/IC2E.2015.74.

[49] WEAVEWORKS. “Docker vs Virtual Machines”, Adresse: https://www.weave.works/blog/a-practical-guide-to-choosing-between-docker-containers-and-vms (besucht am 12. 09. 2021).

[50] Docker, Inc. “Docker Engine overview”, Adresse: https://docs.docker.com/engine/ (besucht am 12. 09. 2021).

[51] ——, “Docker Subscription and Licensing Questions”, Adresse: https://www.docker.com/pricing/faq (besucht am 12. 09. 2021).

[52] Marek Moravcik, Pavel Segec, Martin Kontsek, Jana Uramova und Jozef Papan, “Comparison of LXC and Docker Technologies”, in *2020 18th International Conference on Emerging eLearning Technologies and Applications (ICETA)*, 2020, S. 481–486. DOI: 10.1109/ICETA51985.2020.9379212.

[53] Sakir Sezer, Sandra Scott-Hayward, Pushpinder Kaur Chouhan, Barbara Fraser, David Lake, Jim Finnegan, Niel Viljoen, Marc Miller und Navneet Rao, “Are we ready for SDN? Implementation challenges for software-defined networks”, *IEEE Communications Magazine*, Jg. 51, Nr. 7, S. 36–43, 2013. DOI: 10.1109/MCOM.2013.6553676.

[54] Open Networking Fundation, “Software-defined networking: The new norm for networks”, *ONF White Paper*, Jg. 2, Nr. 2-6, S. 11, 2012.

[55] Adrian Lara, Anisha Kolasani und Byrav Ramamurthy, “Network Innovation using OpenFlow: A Survey”, *IEEE Communications Surveys Tutorials*, Jg. 16, Nr. 1, S. 493–512, 2014. DOI: 10.1109/SURV.2013.081313.00105.

[56] Open Networking Foundation. “OpenFlow Switch Specification”, Adresse: https://opennetworking.org/wp-content/uploads/2014/10/openflow-switch-v1.5.1.pdf (besucht am 12. 09. 2021).

[57] Linux Foundation Collaborative. “Open vSwitch”, Adresse: https://www.openvswitch.org/ (besucht am 12. 09. 2021).

[58] Ben Pfaff, Justin Pettit, Teemu Koponen, Ethan Jackson, Andy Zhou, Jarno Rajahalme, Jesse Gross, Alex Wang, Joe Stringer, Pravin Shelar u. a., “The design and implementation of open vswitch”, in *12th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 15)*, 2015, S. 117–130.

[59] The VyOS Project. “VyOS”, Adresse: https://vyos.io/products/#vyos-router (besucht am 12. 09. 2021).

[60] . “VyOS as docker container journey”, Adresse: https://forum.vyos.io/t/vyos-as-docker-container-journey/6128 (besucht am 12. 09. 2021).

[61] 2stacks. “VYOS on Docker”, Adresse: https://hub.docker.com/r/2stacks/vyos (besucht am 12. 09. 2021).

[62] VyOS maintainers and contributors. “VyOS Hardware requirements”, Adresse: https://docs.vyos.io/en/crux/installation/install.html#hardware-requirements (besucht am 12. 09. 2021).

[63] ——, “Building VyOS”, Adresse: https://docs.vyos.io/en/crux/contributing/build-vyos.html (besucht am 12. 09. 2021).

[64] Paul Jakma. “Quagga Routing Software Suite”, Adresse: https://www.quagga.net/ (besucht am 12. 09. 2021).

[65] Quagga. “Quagga GIT Repository”, Adresse: https://github.com/Quagga/quagga/commits/master (besucht am 12. 09. 2021).

[66] FRR. “FRRouting Project”, Adresse: https://frrouting.org/ (besucht am 23. 08. 2021).

[67] ——, “FRRouting Documentation”, Adresse: https://docs.frrouting.org/en/latest/overview.html (besucht am 12. 09. 2021).

[68] Canonical Ltd. “Ubuntu”, Adresse: https://ubuntu.com/ (besucht am 23. 08. 2021).

[69] The PHP Group. “PHP”, Adresse: https://www.php.net/ (besucht am 23. 08. 2021).

[70] Pluralsight. “Javascript”, Adresse: https://www.javascript.com/ (besucht am 23. 08. 2021).

[71] Proxmox Server Solutions GmbH. “Proxmox VE Command line tools”, Adresse: https://pve.proxmox.com/pve-docs/pve-admin-guide.html#_command_line_interface (besucht am 04. 09. 2021).

[72] Dr. Steve E. Deering und Bob Hinden, *Internet Protocol, Version 6 (IPv6) Specification*, RFC 8200, Juli 2017. DOI: 10.17487/RFC8200. Adresse: https://rfc-editor.org/rfc/rfc8200.txt.

[73] Proxmox Server Solutions GmbH. “Proxmox VE Cluster Manager”, Adresse: https://pve.proxmox.com/wiki/Cluster_Manager (besucht am 04. 09. 2021).

[74] Hal Finney, Lutz Donnerhacke, Jon Callas, Rodney L. Thayer und David Shaw, *OpenPGP Message Format*, RFC 4880, Nov. 2007. DOI: 10.17487/RFC4880. Adresse: https://rfc-editor.org/rfc/rfc4880.txt.